SEMANTIC MODELS AS METRICS FOR KERNEL-BASED INTERACTION IDENTIFICATION

by

Tamara Polajnar

A thesis submitted to
the Faculty of Information and Mathematical Sciences
of the University of Glasgow for the degree of Doctor of Philosophy

Glasgow, June 2010

Abstract

Automatic detection of protein-protein interactions (PPIs) in biomedical publications is vital for efficient biological research. It also presents a host of new challenges for pattern recognition methodologies, some of which will be addressed by the research in this thesis. Proteins are the principal method of communication within a cell; hence, this area of research is strongly motivated by the needs of biologists investigating sub-cellular functions of organisms, diseases, and treatments. These researchers rely on the collaborative efforts of the entire field and communicate through experimental results published in reviewed biomedical journals. The substantial number of interactions detected by automated large-scale PPI experiments, combined with the ease of access to the digitised publications, has increased the number of results made available each day. The ultimate aim of this research is to provide tools and mechanisms to aid biologists and database curators in locating relevant information. As part of this objective this thesis proposes, studies, and develops new methodologies that go some way to meeting this grand challenge.

Pattern recognition methodologies are one approach that can be used to locate PPI sentences; however, most accurate pattern recognition methods require a set of labelled examples to train on. For this particular task, the collection and labelling of training data is highly expensive. On the other hand, the digital publications provide a plentiful source of unlabelled data. The unlabelled data is used, along with word cooccurrence models, to improve classification using Gaussian processes, a probabilistic alternative to the state-of-the-art support vector machines. This thesis presents and systematically assesses the novel methods of using the knowledge implicitly encoded in biomedical texts and shows an improvement on the current approaches to PPI sentence detection.

Acknowledgements

Author's Declaration

I declare that, except where explicit reference is made to the contribution of others, that this dissertation is the result of my own work and has not been submitted for any other degree at the University of Glasgow or any other institution.

Tamara Polajnar

Table of Contents

List of Figures

List of Tables

List of Abbreviations

BEAGLE	bound encoding of the aggregate language environment
GP	the Gaussian processes classifier
HAL	hyperspace analogue to language
IE	information extraction
IR	information retrieval
LDA	latent Dirichlet allocation
LSA	latent semantic analysis
NB	the naïve Bayes classifier
NE	a named entity
NER	named entity recognition
pMKL	probabilistic multiple kernel learning
PPI	a protein-protein interaction
SVM	the support vector machine classifier
TM	text mining
VBpMKL	variational Bayes probabilistic multiple kernel learning

List of Symbols

$\mathbf{A}$	a matrix
$\mathbf{a}_i$	a row vector in a matrix
$\mathbf{A}^T$	the transpose of a matrix
$\mathbf{B}$	a BEAGLE word co-occurrence matrix
$\mathbf{H}$	a HAL word co-occurrence matrix
$\mathbf{H}_L$	a HAL matrix created with window of length L
$\mathbf{H}_l$	a HAL matrix of words co-occurring at distance l
$\mathbf{K}$	a kernel matrix
$\mathbf{X}$	a training data matrix
$\mathbf{x}$	a vector of training data
$\mathbf{w}$	a vector of training weights
w_0	weight vector offset
$\mathbf{y}$	a vector of class labels
D	the number of dimensions in the BEAGLE matrix
K	the number of classes
M	the number of training documents
N	the number of features
W	the set of features
T	the set of target words
$\|T\|$	the number of target words
B	the set of basis words
$\|B\|$	the number of basis words
w_j, t_j, b_j	a word in a set of words W, T, or B

κ	a kernel transformation function
θ	a model parameter, in general the Gaussian kernel parameter
$\boldsymbol{\beta}$	the kernel combination weight vector
β	a weight assigned to a kernel
C	the SVM margin parameter

Chapter 1

Introduction

Proteins are the principal engine enabling chemical reactions in a cell, and, as such, are of great interest to biologists studying life on the molecular level. The cell is the minimal self-reproducing unit and is the vehicle for the transmission of the genetic information in all living species. Contained within the nucleus of the cell is deoxyribonucleic acid (DNA). DNA is a double-stranded polymer that the cell replicates through the separation of these strands. Each strand is a template, which is then used to polymerise a new DNA strand with a complementary sequence. A similar polymerisation process is used to *transcribe* portions of the information held in the DNA into molecules of the closely related polymer, the ribonucleic acid (RNA). RNA is a blueprint for protein synthesis through a more complex process called *translation* (Alberts et al., 2002).

Part of the proteins' functionality depends on their interactions with other cellular components. Understanding these interactions is paramount to the understanding of pathologies, diseases, and treatments. Of particular interest to biologists are interactions between proteins, which are often tracked and represented as a network, such as the one shown in Figure 1.1. A specific protein can be present in different complex sets of interactions that have outcomes with different purposes. Each sequence of interactions is referred to as a *pathway* (Alberts et al., 2002).

The principal observations of interactions are made through biological experiments

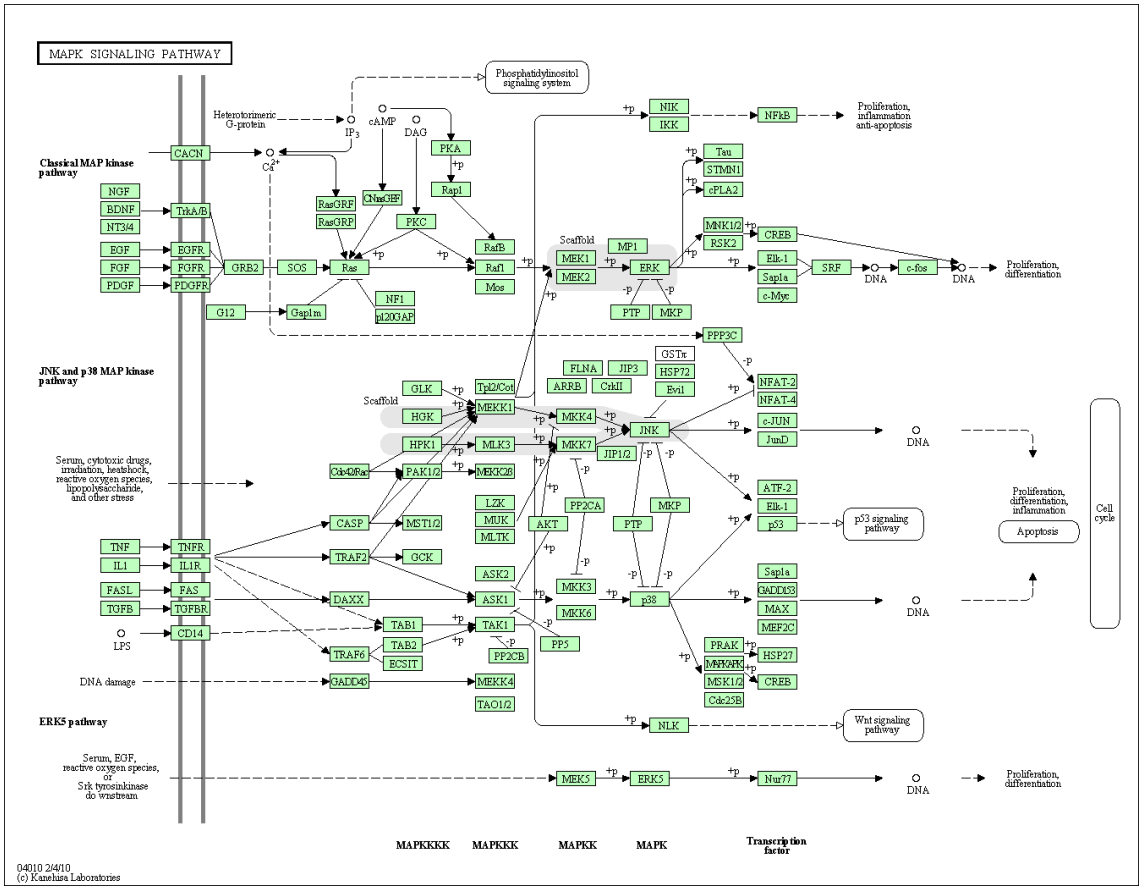


Figure 1.1: An example of a schematic network diagram used by biologists to describe a pathway. This diagram shows the mitogen activated protein kinase (MAPK) signalling pathway for the human species, and is sourced from the KEGG database (Kanehisa et al., 2010). Pathways are a representational construct, a way to visualise a specific set of interactions, and new pathways are being discovered all the time.

(Young, 1998), whose results are reported in peer-reviewed biomedical journal articles. Protein-protein interactions (PPIs) are then found by researchers through various search engines indexing these specific articles; however, current state-of-the-art search engines are not well suited for this task, as *ad hoc* query-based searches are more appropriate for temporary information needs, not persistent ones (Nanas et al., 2009). For research tasks such as pathway construction or population of PPI databases such as KEGG (Kanehisa et al., 2010), MIPS (Mewes et al., 2004), or BIND (Alfarano et al., 2005), PPI extraction becomes a continuous process. Consequently, PPI detection and extraction have become one of the primary goals of biomedical text mining (TM) (Cohen and Hersh, 2005). The aim is to develop applications that will enable habitual PPI searchers to find interactions

without having to specify pairs of proteins or manually scan large amounts of text.

Although analogous systems have been developed for other domains, such as news, biomedical texts offer particular challenges that need to be addressed with tailored tools (Cohen and Hersh, 2005; Albert et al., 2003). For example, the detection of the protein names is a difficult problem due to high degree of synonymy, polysemy, orthographic variation, and novelty due to protein discovery (Hirschman et al., 2002; Subramaniam et al., 2003; Tanabe et al., 2005; Alex et al., 2007; Smith et al., 2008). Protein name recognition is not a necessary step for detection of areas of text that describe interactions (Donaldson et al., 2003; Polajnar et al., 2009b), but for more detailed extraction it is essential (Sekimizu et al., 1998; Friedman et al., 2001; Rebholz-Schuhmann et al., 2007; Rosario and Hearst, 2005; Marcotte et al., 2001; Giuliano et al., 2006; Bunescu et al., 2005).

This thesis introduces a method that improves detection of sentences describing PPIs in biomedical texts. The technique is based on enhancing the state of the art classification-based PPI approaches previously developed (Donaldson et al., 2003; Bunescu et al., 2005; Giuliano et al., 2006; Airola et al., 2008; Erkan et al., 2007). Classification-based methods are usually trained on data labelled by experts. The technique described herein is envisioned as a component of a trainable filtering system, that could be placed on top of a keyword search and could effectively learn from simple annotations provided by a user. For example, a user could indicate whether a sentence describes a PPI or not. Such annotation schemes would be less onerous than ones that requires users to label each protein participating in an interaction and perhaps any other words indicating their relationship. While any pattern recognition or discriminant analysis method could be used for this purpose, the main contribution of this thesis is a novel method that enhances the effectiveness of learning from the labelled examples by incorporating semantic information from unlabelled data; and thus reducing the burden on the user.

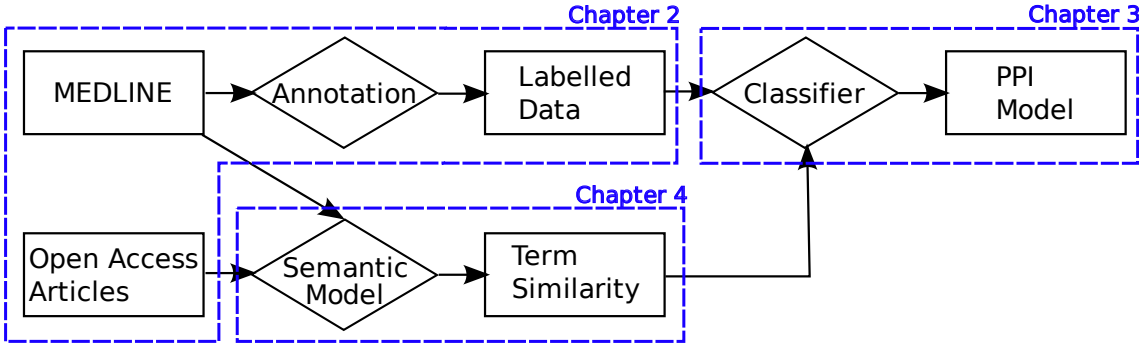


Figure 1.2: An illustration of the method used in this thesis. The background chapters which explain the individual components are highlighted.

1.1 Structure of the document

This thesis is separated into two parts. The background knowledge that underpins the method, as illustrated in Figure 1.3, is contained in Part I, which is organised as follows:

- **Chapter 2** contains the description of PPIs and the challenging aspects of PPI detection. Included here are descriptions of some of the key resources that will be used in this thesis and the literature review of the different methods applied to the this problem.

- **Chapter 3** provides the background knowledge on classification, including the baseline classifier, naïve Bayes (NB) (Lewis, 1998; Nigam et al., 2006); the state of the art support vector machine (SVM) (Cristianini et al., 2002; Bishop, 2006); the proposed probabilistic alternative to the state of the art, the Gaussian process classifier (GP) (Rasmussen and Williams, 2006; Bishop, 2006); and finally a related probabilistic algorithm that can learn a single model from multiple representations of a training example (pMKL) (Damoulas and Girolami, 2008). This chapter also discusses the concept of a *kernel*, which is a transformation of the input space that is an integral part of the SVM, GP, and pMKL algorithms. Fundamentally, it is alterations to the kernel that provide improvements in classification presented in this thesis.

- **Chapter 4** describes two methods of collecting semantic information, which are used to enrich the training data in the classification process. By observing word usage across a large corpus, these methods compute term similarity statistically. This semantic information is used to inform the judgement of PPI sentence similarity by making allowances for word interchangeability as judged by the term similarity scores.

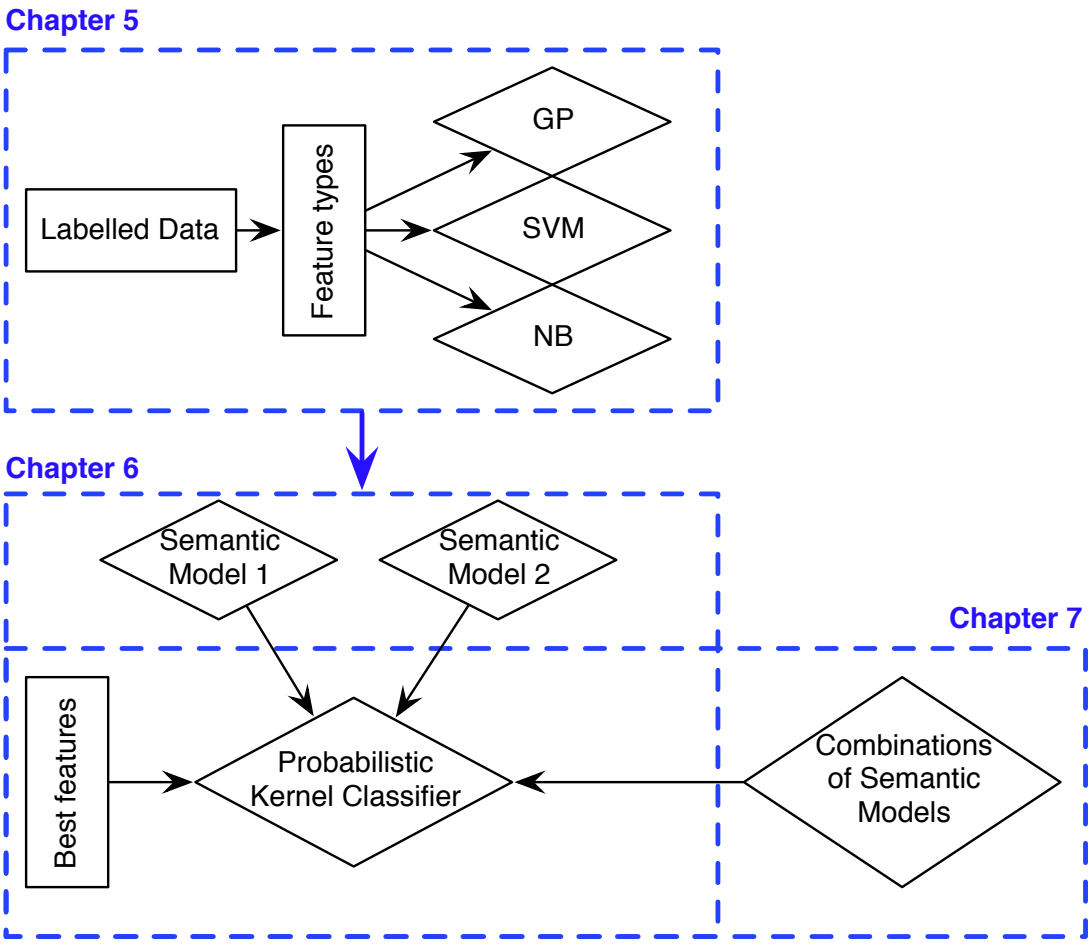


Figure 1.3: An illustration of the experimental chapters of the thesis. The results of the classifier comparison and feature optimisation from Chapter 5 are used in the following chapters to examine the effects of using semantic models, both individually (Chapter 6) and in combination(Chapter 7).

The experiments exploring this approach are contained in Part II:

- **Chapter 5** is a collection of experiments comparing the algorithms described in Chapter 3 on different datasets and with different feature types, kernels, and kernel

settings.

- **Chapter 6** describes the semantic information integration process and the consequent improvements in classification performance. In this chapter the information provided by each of the semantic models described in Chapter 4 is used in classification separately.

- **Chapter 7** shows how this information can then be combined to train a single classifier in order to gain further improvements.

- **Chapter 8** summarises the contributions of this thesis and describes the future directions that will be explored based on the foundations laid herein.

These experimental chapters follow a logical progression and the results detailed within are improved upon in successive stages.

1.2 Thesis statement, hypothesis, and contributions

This thesis aims to explore the ways in which large amounts of unlabelled data can be used to improve the classification of PPI abstracts. The unlabelled data is semantically and topically related to the labelled data, but may not come from the same distribution; therefore impeding the usage of traditional semi-supervised learning methods. A new methodology, which incorporates the semantic similarity of the terms in the unlabelled data into the kernel classification algorithms, is used to improve upon the state-of-the-art results. In order to provide the most challenging baseline, this thesis compares several classifiers, including Gaussian processes, a probabilistic alternative to the popular support vector machines. Contributions from this thesis include an in-depth comparative analysis of several classification methods, datasets, and features types; as well as new methodologies for integrating unlabelled data. Further contributions are detailed below.

The following hypotheses underlie the research in this thesis:

1. *For PPI sentence classification, the GPs are equivalent in performance to the SVMs.* Chapter 5 and Appendix A contain tables and graphs showing in-depth and summarised analysis of the results from a wide range of experiments on these algorithms. The results show that given the right choice of kernel, however, the difference between the algorithms is not statistically significant.

2. *The GPs are suitable for PPI detection because the Bayesian framework of the GPs allows for useful extensions and provides probabilistic output.* Chapter 3 provides a comparison of the theoretical aspects of the algorithms. It is demonstrated that the Bayesian framework within which the GPs are constructed can be used to make derivative algorithms tailored for situations that may arise in biomedical TM. For example, the multiclass GP can be applied to datasets where each interaction is subdivided by type, such as in Rosario and Hearst (2005). This application was demonstrated in Polajnar et al. (2009b). Furthermore, as is explained in Chapter 2 and in Alex et al. (2008b), relevance annotation for PPI sentences can be a complicated problem that sometimes leads to intrinsic disagreements between annotators. These contested annotations can sometimes be costly to disambiguate, or inherently ambiguous. In this case the multiexpert GP (Rogers, S., Girolami, M., and Polajnar, T., 2009) can be used to learn from multiple expert annotators and provide assessments of the quality of the individual annotators, enabling automatic disambiguation.

3. *The GPs are suitable for an interactive environment because they have probabilistic output.* Chapter 3 shows that the GP probabilistic output gives a wider diversity of values (between 0 and 1) than the SVM approximation of probabilistic output. In that way, the GPs are giving a user more information about how confident the model is in assigning an instance to a specific class.

4. *Semantic knowledge from word similarity models can improve kernel classification.* Chapter 6 shows that integrating word similarity information collected from unla-

belled data can improve the accuracy of PPI classification. This improvement can vary depending on the choice of features and semantic models.

5. *Using combinations of semantically enriched kernels can lead to an increase classification performance. In addition, insight into the information contained in the training data can be gained by estimating the contributions of individual kernels.* Experiments in Chapter 7 show how combinations of the same feature space enriched by different semantic information can be used to train a classifier in order to produce an even larger improvement in classification accuracy.

1.3 Supporting publications

This thesis has produced a number of peer-reviewed publications to which the interested reader is referred:

- Polajnar, T., Rogers, S., and Girolami, M. (2009b). Protein interaction detection in sentences via Gaussian processes: A preliminary evaluation. *International Journal of Data Mining and Bioinformatics*. To appear

  Abstract. Classification methods are vital for efficient access of knowledge hidden in biomedical publications. Support vector machines (SVMs) are modern non-parametric deterministic classifiers that produce state of the art performances in text mining, and across other disciplines, while reducing the need for feature engineering. In this paper we offer a much needed evaluation of the Gaussian Process (GP) classifier, as a non-parametric probabilistic analogue to SVMs, which has been rarely applied to text classification. To this end, we provide an extensive experimental comparison of the performance and properties of these competing classifiers on the challenging problem of protein interaction detection in biomedical publications. Our results show that GPs can match the performance of SVMs without the need for costly margin parameter tuning, whilst offering the advantage of an extendable probabilistic framework for text classification.

- Polajnar, T., Rogers, S., and Girolami, M. (2009a). Classication of protein interaction sentences via Gaussian processes. In *Proceedings of 4th IAPR International*

Conference, Pattern Recognition in Bioinformatics, pages 282–292. Springer Verlag

> **Abstract.** The increase in the availability of protein interaction studies in textual format coupled with the demand for easier access to the key results has lead to a need for text mining solutions. In the text processing pipeline, classification is a key step for extraction of small sections of relevant text. Consequently, for the task of locating protein-protein interaction sentences, we examine the use of a classifier which has rarely been applied to text, the Gaussian processes (GPs). GPs are a non-parametric probabilistic analogue to the more popular support vector machines (SVMs). We find that GPs outperform the SVM and naïve Bayes classifiers on binary sentence data, whilst showing equivalent performance on abstract and multiclass sentence corpora. In addition, the lack of the margin parameter, which requires costly tuning, along with the principled multiclass extensions enabled by the probabilistic framework make GPs an appealing alternative worth of further adoption.

- Polajnar, T. and Girolami, M. (2009a). Application of lexical topic models to protein interaction sentence prediction. In *NIPS 2009 Workshop on Applications for Topic Models: Text and Beyond*, Whistler, Canada

  > **Abstract.** Topic models can be used to improve classification of protein-protein interactions (PPIs) by condensing lexical knowledge available in unannotated biomedical text into a semantically-informed kernel smoothing matrix. Detection of sentences that describe PPIs is difficult due to lack of annotated data. Furthermore, sentences generally contain a small percentage of the features, thus leading to sparse training vectors. By exploiting contextual similarity of words we are able to improve the classification performance. This contextual data is gathered from a large unannotated corpus and incorporated through a semantic kernel. We use Hyperspace Analogue to Language (HAL) and Bound Encoding of the Aggregate Language Environment (BEAGLE) semantic models to create the kernels. The modularity of the method lends itself to further exploration along several different avenues including experimentation with any number of word and topic models.

- Polajnar, T. and Girolami, M. (2009b). Semi-supervised prediction of protein interaction sentences exploiting semantically encoded metrics. In *Proceedings of the 4th IAPR International Conference, Pattern Recognition in Bioinformatics*, pages 270–281. Springer Verlag

Abstract. Protein-protein interaction (PPI) identication is an integral component of many biomedical research and database curation tools. Automation of this task through classication is one of the key goals of text mining (TM). However, labelled PPI corpora required to train classifiers are generally small. In order to overcome this sparsity in the training data, we propose a novel method of integrating corpora that do not contain relevance judgements. Our approach uses a semantic language model to gather word similarity from a large unlabelled corpus. This additional information is integrated into the sentence classication process using kernel transformations and has a re-weighting effect on the training features that leads to an 8% improvement in F-score over the baseline results. Furthermore, we discover that some words which are generally considered indicative of interactions are actually neutralised by this process.

- Polajnar, T., Damoulas, T., and Girolami, M. (2010). Protein interaction sentence detection using multiple semantic kernels. Under review for the International Journal of Systems Science special issue on Integrative Genomics

Abstract. Detecting protein-protein interactions in biomedical publications is a challenging and unresolved pattern recognition problem. In this work we propose a novel data integration approach that utilises semantic kernels, which are created from statistical information gathered from large amounts of unlabelled text, and then fused into an overall composite classification space. We show statistically significant improvements in recognition rates and receiver operating characteristic (ROC) scores over previously published results on a well known labelled collection of abstracts, while automatically inferring the most discriminative resolution levels for constructing the semantic information sources.

- Rogers, S., Girolami, M., and Polajnar, T. (2010). Semi-parametric analysis of multi-rater data. *Statistics and Computing*, 20:317–334. 10.1007/s11222-009-9125-z

Abstract. Datasets that are subjectively labeled by a number of experts are becoming more common in tasks such as biological text annotation where class definitions are necessarily somewhat subjective. Standard classification and regression models are not suited to multiple labels and typically a pre-processing step (normally assigning the majority class) is performed. We propose Bayesian models for classification and ordinal regression that naturally incorporate multiple expert opinions in defining

predictive distributions. The models make use of Gaussian process priors, resulting in great flexibility and particular suitability to text based problems where the number of covariates can be far greater than the number of data instances. We show that using all labels rather than just the majority improves performance on a recent biological dataset.

Part I

Background

Chapter 2

Protein Interaction Extraction

Protein-protein interaction (PPI) extraction is a key application of text mining to biological texts (Cohen and Hersh, 2005; Krallinger et al., 2005). This area of research is strongly motivated by the needs of biologists investigating sub-cellular functions of organisms (Alex et al., 2008a; Hirschman et al., 2005a). Proteins are biological entities that are generated from the genes constituting the deoxyribonucleic acid (DNA). They perform essential functions in cells by interacting with each other and other cellular components. Consequently, the study of proteins is integral to the understanding of organism function and disease treatment (Alberts et al., 2002; Albert et al., 2003).

PPIs reported in biomedical journals are detected by large-scale biomedical experiments, such as the yeast two-hybrid (Young, 1998). The substantial number of results produced by these experiments, combined with the ease of access to the digitised publications provided by the various publisher portals, has increased the number of results made available each day (Roberts, 2006). Searching for just a single well-studied pathway can lead to thousands of results. For example, the query *mapk pathway* (shown in Figure 1.1) on PubMed[1], a search engine which indexes MEDLINE[2], produces 9,389 results[3]. In addition, the number of citations in MEDLINE has been growing steadily,

[1] http://www.ncbi.nlm.nih.gov/pubmed/

[2] MEDLINE (http://www.nlm.nih.gov/pubs/factsheets/medline.html) is a comprehensive, curated database of biomedical citations from 1949 until the present.

[3] In January, 2010

with at least 600,000 new entries being added in each of the years between 2005 and 2009[4] and totalled over 17,000,000 citations by the end of 2009. MEDLINE contains freely accessible abstracts for a large subset of the articles. These have become a major resource for biomedical text mining and are the basis of several key corpora used to study PPIs (Krallinger et al., 2008a). Although further results are available in the full text articles, many of the journals are only accessible through on-line subscription-only portals or through pay-per-download articles. However, recently, more and more articles are becoming available through open access publishing (Cockerill, 2008).

On one hand, the data has become more accessible; on the other, search engines can return thousands of relevant results based on a keyword search, making highly specific information difficult to locate (Hersh, 2005; Roberts, 2006; Cohen and Hersh, 2005). Curated databases, focusing on particular research tasks or organisms, have been developed to allow for quicker and more targeted access. Nevertheless, the curators still have a difficult task of finding the results required to populate the databases (Hirschman et al., 2005a; Alex et al., 2008a). The rest of this chapter describes the structure of PPIs and the problem of locating them in text, previous relevant methods applied to this task, and required background knowledge on the available resources and evaluation.

2.1 PPI sentences

PPIs are defined as a triple containing two protein entities and an action word describing the relationship linking them. The protein names are a set which is continuously expanding through new discoveries, typographic variations, and synonym generation; on the other hand, the words describing the actions belong to a set of finite size. These words are usually morphological variations of key interaction words such as *activate* or *interact*. For example, the different variation of the root *bind* can be:

- the verb, *PTN1 binds with PTN2*;

[4] http://www.nlm.nih.gov/bsd/bsd_key.html

- the present (*-ing*) or past (*-ed*) participial adjectives, *binding PTN1 interacts with PTN2*;
- or nouns preceded by an adjectival phrase containing proteins, *a PTN1-PTN2 binding*.

Likewise, there can be great variations in sentence structure. Figure 2.1 shows examples of different ways an interaction can be described within sentences.

1. This work shows that single and double Ala substitutions of His18 and Phe21 in IL-8 reduced up to 77-fold the binding affinity to IL-8 receptor subtypes A (CXCR1) and B (CXCR2) and to the Duffy antigen.
2. This study demonstrates that IL-8 recognizes and activates CXCR1 CXCR2, and the Duffy antigen by distinct mechanisms.
3. CXCL8 (also known as IL 8 [IL-8]) activates CXCR1 and CXCR2 to mediate neutrophil recruitment and trigger cytotoxic effect at sites of infection.
4. Interleukin 8 (IL 8 [IL-8]) appears to have a fundamental role in regulating neutrophil localization in ischemic tissues through binding CXCR1 CXCR2 receptors, which show major expression on neutrophils.
5. This rescue effect could be blocked by antibodies to the IL 8 [IL-8] receptor CXCR1 but not by CXCR2, suggesting that normal urothelial cells normally have IL 8 [IL-8] autocrine or paracrine activity for survival and growth mediated by CXCR1.

Figure 2.1: Several different sentences describing interactions between *IL-8* and *CXCR1* and *CXCR2*.

The three elements of a PPI can appear in text in an assortment of configurations. They may be found in one sentence or across a paragraph, linked by referring expressions. They can even be described in parts across the whole document, in a table, in an appendix, or in an image representing a network of interactions.

PPIs are then, by nature, difficult to describe using keyword search, especially if one is trying to find new interacting pairs. Therefore, many articles may need to be read in order to locate and verify a particular interaction. This painstaking search is often performed by PPI database curators and researchers. For example, a researcher looking to build an accurate map of an interaction pathway can start from a single protein. A search for *MAPK (mitogen activated protein kinase)* results in 20,845 citations on a variety of

subjects for many different species. By restricting the search to the *human* species we can half the results to 10,429. Restricting with a selection of other keywords such as *interaction*, *activation*, *binding* leads to further, possibly overlapping lists, with 826, 7,361, and 3,088 results respectively. Therefore, query refinement can still result in a large number of documents that could be relevant to a biologist searching for all possible proteins linked to *MAPK*.

The ultimate goal of text mining is to automate this process of information extraction to a degree that would allow accurate automatic creation of customised databases or interaction network maps (Donaldson et al., 2003; Jenssen et al., 2001; Chen and Sharp, 2004; Rzhetsky et al., 2004; Alex et al., 2008a,c); however, compromises are currently being made by concentrating on results that are reported in text, mainly from the freely available abstracts, and usually in a single sentence (Ding et al., 2002; Oda et al., 2008).

Moreover, the process of interaction detection is cyclical. In the process of finding the searcher creates more information for the training of new assistance systems. For example, the curated databases themselves are an important resource for generating the PPI interaction detection software. Database of Interacting Proteins (DIP) (Salwinski et al., 2004; Marcotte et al., 2001; Donaldson et al., 2003), HIV-1 Human Protein Interaction Database (HPID) (Fu et al., 2009; Rosario and Hearst, 2005), and IntAct (Kerrien et al., 2007; Hakenberg et al., 2006) have all been used to generate data sets.

2.1.1 Types of interactions and interaction indicators

The actual words that constitute the three components of an interaction are not the only characteristics that distinguish sentences that contain PPIs. The discriminative words also vary depending on the topic and type of interaction that is being considered.

For example, in a corpus constructed from the abstracts referenced in DIP, Marcotte et al. (2001) found that the following words appear most frequently in the abstracts describing protein interactions in yeast: *complex, interaction, two-hybrid, interact, proteins, protein, domain, interactions, required, kinase, interacts, complexes, function, essential,*

binding, component. Many of these words are not the verbs directly describing the interaction, but talk about protein complexes or the yeast two-hybrid method for finding interactions and complexes. The words were chosen because their occurrence in abstracts that describe protein interactions is higher than the average occurrence across all the abstracts on the topic of yeast.

Similarly we can see from another data set, which contains both positive and negative examples (Bunescu et al., 2005), that the top most frequent words that occur in positive sentences can be similar in some cases to the top words from negative sentences. The most frequent words in the positive data are: *binding, protein, receptor, interaction, il, beta, domain, complex, cells, human, cell, kinase*. While for the negative they are: *protein, receptor, cell, binding, cells, human, proteins, il, transcription, interaction, domain, expression*. Thus words such as *protein* and *binding* are poor class discriminators for this corpus.

From a data set that is based on the HPID database (Rosario and Hearst, 2005), in which the interactions are also classified by type, we can see that different words occur in sentences that describe different kinds of interactions. In Table 2.1 we can see that the lists vary greatly, although some proteins such as *Tat* (*trans-activator of transcription*) occur frequently throughout the corpus. This database focuses on a small number of proteins that regulate the HIV virus. The proteins appear to interact in specific ways, and thus only occur in some of the lists. In addition words such as *apoptosis* and *growth* also reflect the functions of these proteins.

2.1.2 Protein names

Biomedical words do not belong to the general language lexicon and have a different morphology including capitalisation, punctuation, and alphanumeric sequences. For example, *MAPK, ERK1, cAMP, MIP-1alpha , CycT1*, and *hHR23A* which are defined in Figure 2.2. However, there are numerous idiosyncrasies not only in the protein and gene names themselves, but also in the way in which they are used by authors, that make the task of their

activates	binds	interacts with	incorporates	enhances	synergizes with
tat	tat	tat	vpr	tat	tat
erk	protein	tfiih	udg	pkr	bfgf
cells	hiv	rev	gag	cells	ks
rsv	gag	transcription	proteins	activation	alpha
induced	rev	nup	virions	nf	cells
activation	tsg	hiv	viral	apoptosis	hiv
hiv	al	rna	hiv	protein	lesions
mapk	domain	sp	hsp	cyclin	al
production	interaction	protein	gst	cdk	growth
protein	tfiih	ii	presence	associated	endothelial

Table 2.1: Examples of frequent words for several classes from the BioText corpus collected by Rosario and Hearst (2005) from the HIV-1 Human Protein Interaction Database. The word strings have been normalised to lower case and exclude any numbers and symbols. High-frequency stop-words have also been disregarded. The number of sentences in each of the above classes is given in the parenthesis: activates (119), binds (417), interacts with (162), incorporates (68), enhances (53), synergizes with (104).

identification more difficult than in the news domain.

1. *MAPK* - any one of a family of proteins, called *mitogen activated protein kinases*, which respond to extracellular signals (Chang and Karin, 2001).

2. *ERK1 - extracellular signal-regulated kinase 1* is a specific MAPK.

3. *cAMP - cyclic adenosine monophosphate*, a nucleotide responsible for inter-cellular signalling and has a role in cancer cell signalling (Abramovitch et al., 2004).

4. *MIP-1alpha - macrophage inflammatory protein-1-alpha*, a protein that causes an inflammatory response (Sherry et al., 1988).

5. *CycT1 - Cyclin T1* is involved in viral gene expression in HIV virus (Okada et al., 2009).

6. *hHR23A* - is a human protein involved in DNA repair (Hsieh et al., 2005).

Figure 2.2: Examples of some biological named entities.

Although proteins and genes are different biological entities, their detection requires similar named entity recognition (NER) systems. Therefore in this section we will discuss the history of NER systems for both proteins and genes as they developed from the late 90s until the present.

2.1.2.1 The trouble with protein names

New proteins are being discovered all the time and being assigned new names, consequently no manually curated list of proteins is ever complete. While some guidelines

for naming proteins exist and newer publications include unique identifiers from protein databases, proteins are traditionally given temporary or memorable synonyms (*e.g. sonic hedgehog*) (Pearson, 2001; Hanisch et al., 2003).

Protein names are often different from nouns which are considered to be named entities in newswire corpora for a variety of reasons (Hirschman et al., 2002; Subramaniam et al., 2003; Cohen and Hersh, 2005; Tanabe et al., 2005; Alex et al., 2007; Smith et al., 2008). Some of these include the following:

- Proteins generally have a number of synonyms and these are ever increasing. For example the protein *ERK* is also known by the following names: *Swiss-Prot: P29323.5, RecName: Full=Ephrin type-B receptor 2; AltName: Full=Tyrosine-protein kinase receptor EPH-3; AltName: Full=DRT; AltName: Full=Receptor protein-tyrosine kinase HEK5; Short=ERK; AltName: Full=Tyrosine-protein kinase TYRO5; AltName: Full=Renal carcinoma antigen NY-REN-47.* However *ERK* is also frequently used to denote *extracellular signal-regulated kinases* and is often used as a synonym for the *Mitogen activated protein kinase* (*MAPK*).

- Article authors use preferred, easy to remember names, instead of proposed standardised symbols, exacerbating polysemy. These names can sometimes be the same as common English words, such as *frazzled, ran, 18 wheeler, etc.* For example, there are 50 proteins who share the short name, *asp*, which is also an English word for a viper.

- Proteins cannot be distinguished only by orthography from genes and other biomedical entities. For example, a molecular function *ATP-binding* or a cell type *HeLa* both have the same unusual capitalisation and hyphenation patterns which characterise proteins. The official guidelines also advocate naming of proteins after the generating genes, however, the protein names should be italicised in text. Unfortunately, the typographic information is not usually available when plain text is being processed.

- Proteins names are sometimes nested or have boundaries that are difficult to define. *Mitogen activated protein kinase* or *MAPK* is part of a cascade of interactions where each protein is phosphorylated by another kinase, leading to a nested naming scheme. Thus, *MAPK kinase (MAPKK)* and *MAPK kinase kinase (MAPKKK)* are also proteins. It is obvious that if the entire protein name is written, a nested entity could be annotated. However, it is also unlikely that *MAPKKK* would be broken up and annotated for each of the component kinases.

  Similarly, proteins that occur in several organisms can be prefixed with a single letter describing the species or species group, *e.g. Mammalian target of rapamycin (mTOR)* or *peptidylglycine alpha-hydroxylating monooxygenase (dPHM)*, which is a *PHM* in the *Drosophila melanogaster* (fruit fly). *Humly9* is only listed as *Ly9* in the human protein database (Smith et al., 2008).

- The biological entities that protein/gene names refer to have multiple states. For example, there are different mutations of a gene, or a protein that can be in a phosphorylated state. *pERK, ERKpp, phosphoERK* are all different ways biologists may refer to a *phosphorylated ERK* protein. As it is merely a different state of the protein, these names will not explicitly occur in the protein database.

- Multi-word names can be written using many different arrangements of capitalisation, spacing, hyphenation, and other orthographies. This can lead to a difficulty in exact matching of the strings.

2.1.2.2 Automatic recognition of protein names

Named entity recognition (NER) is a key aspect of information extraction (IE). In news and email corpora IE concentrates on finding people, locations, companies, and time events in order to extract key historical events. In biological texts NER is a more difficult problem due to the ambiguity and structure of the domain terminology (as outlined in Section 2.1.2.1). Initial protein name identification systems had the reported F-measure

(the evaluation measures are described in depth in Section 2.2.3) of around 0.75 while the contemporary NER systems in the newswire domain were at 0.95 (Krauthammer et al., 2000; Hirschman et al., 2002; Tanabe and Wilbur, 2002; Cohen and Hersh, 2005). These initial results were difficult to compare due to lack of standardised training data.

In order to encourage development in this field, the *BioCreative: critical assessment of information extraction for biology* competition was run in 2004. The organisers provided training data to the participating teams for two tasks: gene name recognition and interaction extraction. The first task was subdivided into two: gene mention (Task 1A) and gene normalisation (Task 1B). In Task 1A, gene names needed to be identified in text, while in Task 1B those names also had to be linked to the correct entities in the specific organism gene databases. The highest scoring teams achieved the balanced F-measure of 0.83 on Task 1A. Task 1B was divided by organism and the top results were 0.92 for the yeast, 0.82 for the fly, and 0.79 for the mouse data (Hirschman et al., 2005a).

A second round of BioCreative was held a few years later. The format was similar, except that Task 1B focused on human genes and proteins. The F-measure on Task 1A improved to 0.87, while on Task 1B for the human data it was 0.81. Moreover, the organisers also tried combinations of the submission results to estimate results that ensemble learning could achieve if it was based on the assessed systems. For Task 1A, this estimated composite classifier achieved 0.905, while for Task 1B the best F-measure was 0.92. These results show that there is still room to create better predictive models using the available training data (Krallinger et al., 2008b).

Alex et al. (2008b) gives statistics on inter-annotator agreement for different types of biomedical entities. In fact, protein name annotation produced the highest agreement on their corpus at F-measure of 0.92, while some rarely occurring entities were more difficult to annotate. Depending on further segmentation of the protein entity, such as by the organism (Hirschman et al., 2005a), the inter-annotator agreement statistics may point to an upper limit of performance for this problem.

2.2 Evaluation of automatic prediction

In the next section, we will examine the automatic detection of protein-protein interactions. In order to facilitate method comparison there is a necessary digression into the evaluation measures that are used to measure the effectiveness of a specific method.

Most natural language processing systems are made of a series of components. Consequently, the evaluation process can concentrate on certain individual components or the whole (Cole, 1997, Chap. 13). The examination of the individual constituents can be accomplished using a series of standard measurements which provide a numerical or graphical performance summary (Van Rijsbergen, 1979; Buckley and Voorhees, 2000; Lewis, 1995; Hersh, 2005; Manning et al., 2008). On the other hand, the evaluation of the system as a whole also requires a more subtle examination of the usability and usefulness (Donaldson et al., 2003; Alex et al., 2008a; Hersh, 2008; Hersh and Hickam, 1998; Hersh, 2005).

Accurate comparison of systems or components requires two things: a standard freely-available dataset, and a full specification of the measures used (Hersh, 2005). If these two standards are available, systems can be easily evaluated against each other; however, for both protein name recognition (Section 2.1.2.2) and PPI detection (Section 2.3) the standard datasets have become available only recently (Cohen and Hersh, 2005; Hersh, 2005).

Standard datasets provide a description of a task, through multiple examples, with human annotations describing the contents in a way which is interpretable by a machine. In information retrieval (IR) the datasets normally contain *relevance judgements*. These are boolean indicators labelling whether a document contains information relevant to a certain query (Manning et al., 2008; Hersh and Voorhees, 2009). The idea carries into more complex information extraction tasks. Annotations can be applied to each word in a corpus, labelling them as belonging to any number of classes. For example, for full sentence parsing one may need to know the type of each word, so the standard corpus would indicate whether a word belonged to the class of verbs, nouns, or other parts of speech

(POS). In PPI corpora, the simplest annotation indicates whether an area of text, such as a sentence or a paragraph, contains an interaction. More complex schemes will specify each of the proteins, the linking interaction words, and perhaps even the dependency trees. The annotation scheme and its complexity dictate the possible uses of the dataset.

2.2.1 Annotation

Biomedical annotation is a fundamentally difficult task because it requires both grammatical and extensive biomedical knowledge. For example, the GENIA corpus annotations were developed in conjunction with biologists, and the annotations were validated by an independent group of biologists (Collier et al., 1999). The TREC Genomics IR collection was annotated by scientist with at least a PhD in biology (Hersh and Voorhees, 2009). Likewise, the annotation of the ITI TXM corpus "was performed by a group of nine biologists, all qualified to PhD level in biology, working under the supervision of an annotation manager (also a biologist) and collaborating with a team of NLP researchers" (Alex et al., 2008b).

For PPIs, the most basic annotation requires a judgement on whether an area of text contains the interaction. In some corpora, there exists only an indication of whether there is an interaction in the abstract (Donaldson et al., 2003). In other more detailed corpora, there may be a label showing which specific sentence contains the interaction (Craven and Kumlien, 1999; Bunescu et al., 2005). Other corpora are also marked with the proteins, and perhaps even full parses of the sentences (Pyysalo et al., 2007; Collier et al., 1999; Kim et al., 2003).

Evidence from corpora that were partially or fully annotated by multiple annotators shows that disagreements arise frequently (Hirschman et al., 2002, 2005a; Wilbur et al., 2006; Alex et al., 2008a). For example, Alex et al. (2008a) found that the F-score of the inter-annotator agreement was 0.85 for protein names, 0.88 for protein name normalisations, 0.65 for PPIs.

Figure 2.3 shows the types of difficulties that can arise by observing one sentence

from the AImed corpus (Bunescu et al., 2005).

This work shows that single and double Ala substitutions of His18 and Phe21 in < **prot**> **IL - 8** </**prot**> reduced up to 77 - fold the binding affinity to <**prot**> < **p1 pair=1** > <**p1 pair=2** > <**p1 pair=3** > <**prot**> **IL - 8** </**prot**> </**p1**> </**p1**> </**p1**> **receptor subtypes A** </**prot**> (<**p2 pair=1** > <**prot**> **CXCR1** </**prot**> </ **p2**>) and B (<**p2 pair=2** > <**prot**> **CXCR2** </**prot**> </**p2**>) and to the <**p2 pair=3** > <**prot**> **Duffy antigen** </**prot**> </**p2**> **.**

Figure 2.3: An example of an annotation error in the AImed dataset.

This sentence demonstrates multiple inconsistencies in annotation. Firstly, the proteins in this sentence are *IL-8, IL-8 receptor subtype A (CXCR1), IL-8 receptor subtype B (CXCR2)*, and *Duffy antigen*. Due to the way the sentence is constructed, it is difficult to precisely annotate the expressions describing subtypes. Furthermore, the interaction participants, although correct, have been erroneously annotated. The main interactor should be the first *IL-8* that is mentioned, and not the one that is nested inside the subtype named entity. The given annotation shows a protein that is interacting with its own synonym.

In addition, Hirschman et al. (2002) argue that the process of annotation, as is needed for machine learning and is common for natural language tasks, may not be natural for biologists because it requires tagging data that may not be relevant to the task. Thus, for named entities, proteins that are not part of the interaction are just as important for machine learning purposes as are the ones that are part of the interaction; however, for biological research purposes, only the relevant information is important. Similarly, Krallinger et al. (2008b) found that systems returned relevant sentences that were not annotated by the curators, which led to deflated evaluated performances by the systems. This confusion about relevance is also reflected in the way many of the standard available corpora are compiled and annotated.

2.2.2 Standard corpora

There are many biomedical corpora covering diverse topics. There are several corpora annotated for PPIs; however, their design targets them for specific domains (Hersh, 2005). For example, in order to build a strong model for identification of sentences that describe

PPIs, examples of both positive and negative classes are needed. However, many of the PPI data sets, such as LLL (Cussens and Nédellec, 2005) and BioInfer (Pyysalo et al., 2007), only provide examples of positive sentences. These data sets are engineered for algorithms that learn the grammatical patterns that consistently occur in PPI sentences. Hence, the corpora are designed for a certain formulation of the problem, and in turn this may limit them for use with a certain family of algorithms.

Some of the standard PPI corpora that are used in this thesis are described below:

- AImed (Bunescu et al., 2005) has emerged as one of the main standard corpora for PPI detection. It consists of abstracts that contain PPI interactions, and have been annotated for proteins with a scheme that distinguishes the interacting protein pairs. It is, therefore, possible to separate the corpus into a data set that contains positive and negative examples. This can be done in two ways. For example, Bunescu et al. (2005), Erkan et al. (2007), and Airola et al. (2008) separate the corpus into pairs of proteins, using the manually annotated protein entities. Interacting pairs are then used as positive training examples, while any two proteins, that occur in the same sentence and do not interact, constitute the negative data.

  The other approach, and one which is employed in this thesis, is to consider the sentences that contain interactions as positive examples, and the ones that do not, as negative. The reformulation of the problem has several advantages. This task is simpler to annotate than the full PPI, which would allow for faster production of training data. Feature extraction does not require sentence parsing or preprocessing in a way that may be sensitive to annotation errors presented in Figure 2.3. The simpler classification task is likely to lead to better predictions of where the PPIs are located, thus while it does not give the full PPIs it might be more useful in a curation pipeline where the results need to be checked by humans (Albert et al., 2003). When processed in this way, the data set contains 614 positive and 1366 negative sentences.

- The PreBIND corpus (Donaldson et al., 2003) is a set of 693 abstracts that contain interactions and 399 which do not. The proteins and the interactions are not manually annotated in the data. This data was used to train a support vector machine to aid in the curation of the BIND database (Alfarano et al., 2005). This data set is unusual due to the higher percentage of positive examples.

- The MIPS data (Craven and Kumlien, 1999) was created from the curated examples contained in the Münich Information Center for Protein Sequences (MIPS) Comprehensive Yeast Genome Database (CYGD) (Mewes et al., 2004). This is a noisy data set that was generated automatically by searching MEDLINE abstracts for co-occurrence of known interacting proteins from the database. This methodology leads to a high probability of false negative labelling of proteins interactions involving entities that are outside of the database, as well as false positive labelling of sentences that are merely a co-occurrence of two proteins and not a description of an interaction. The MIPS data set contains 498 positive and 5,728 negative abstracts encountered during the data collection phase. Within these, there are 46,931 automatically annotated sentences, out of which there are 41,475 negative and 5,456 positive. This is the largest data set available, even if it is noisy. Due to the inclusion of the negative abstracts, the balance of positive to negative examples reflects more accurately the sparsity of PPIs in MEDLINE.

- BioCreative PPI[5] data was compiled from 1000 randomly chosen sentences from the BioCreative I Task IA (Hirschman et al., 2005b; Yeh et al., 2005) and annotated for part of speech tags, gene/protein names, and interaction-indicating verbs. It contains 173 positive sentences describing a total of 255 interactions.

- Biotext (Rosario and Hearst, 2005) is a multi-class PPI data set that was created from the HIV-1 Human Protein Interaction Database (HPID) (Fu et al., 2009). There are 25 classes of interactions with supporting evidence drawn from the re-

[5]http://www2.informatik.hu-berlin.de/ hakenber/corpora/

ferring papers. The class size varies from 39 to 416 sentences from full text papers, to total 3,025 examples.

Other corpora used in this thesis include the GENIA corpus (Kim et al., 2003) and the collection of open access articles (OAA) from BioMed Central[6]. These are used as large corpora for gathering semantic information. GENIA has almost 450,000 words collected from MEDLINE abstracts tagged with *human, blood cells*, and *transcription factor*. The OAA corpus contains over 54 million words from full text publications in dozens of journals covering biomedical topics.

Some of the other relevant corpora that did not fit the experiments in this thesis, or were not available at the time of the research are GENIA Pathway corpus (Oda et al., 2008), BioCreative II (Krallinger et al., 2008b,b), BioInfer (Pyysalo et al., 2007), and LLL (Cussens and Nédellec, 2005). Most notably, there is the ITI TXM corpus (Alex et al., 2008b), which is closest to an ideal PPI corpus, although it is not freely available. Aat 217 full-text documents, the ITI TXM PPI corpus is one of the largest. While BioInfer may have a richer ontology for entity and relationship tags, TXM contains important information on the negative space of the documents, the rejected papers and the non-interacting sentences. Created in conjunction with a company undertaking biomedical curation, this data attempts to model the process fully, even recording the initial search terms and the document selection process. This kind of information is useful, not only for modelling the data selection process, but likewise for obtaining extra unlabelled data, from the same distribution, for use with semi-supervised algorithms. The corpus was annotated for PPIs and tissue expression, and includes several relevant entity types. The PPI component consists of full text open access papers in XML format. By using full text, the curation process and the distribution of the PPIs are modelled more realistically and include more detailed experimental information. Alex et al. (2008b) describes the annotation process in detail including the inter-annotator agreement scores, although they do not mention if the annotators themselves are tracked in the released version of the corpus. The knowledge

[6]http://www.biomedcentral.com/info/about/datamining/

of annotators (*i.e.* who performed the labellings) would open up the corpus for further research. For example, Rogers et al. (2010) designed an algorithm that can learn from multiple annotators, produce evaluations of annotator performance, and therefore be used to disambiguate difficult annotation processes. In fact, the inter-annotator agreement for entities in the ITI TXM corpus varies from F-measure (Section 2.2.3) of 0.60 to 0.91, demonstrating the difficulties in annotation of certain biomedical entities. The interaction annotation agreement, in general, is higher, especially when the properties of interactions, such as whether it is positive (0.99) or negative (0.90), are concerned.

2.2.3 Evaluation measures for performance comparison

Once there is an established standard corpus for a particular problem, then it is possible to compare algorithm performance against each other. However, the choice of evaluation metrics is important, as different metrics measure different aspects of the algorithms. Initially it seems straight forward that the algorithm that assigns more correct labels would be the better performing one, but it turns out that the balance between the number of correct labels and the type of class assignment is essential. For example, the simplest measure is the one that tells us what percentage of the data was assigned the proper labels. Nevertheless, this measure, called *accuracy*, does not always give a faithful description of the algorithm performance. In particular, if a dataset has 80% negative examples, an algorithm can score 80% accuracy simply by guessing negative for every data point.

For this reason, natural language processing algorithms are mainly evaluated using the *F-measure* (Van Rijsbergen, 1979). It is defined terms of *true positives (tp), false positives (fp), true negatives (tn), false negatives (fn)* (Figure 2.4), and from those, by *precision* and *recall*:

$$precision = \frac{tp}{tp + fp} \quad recall = \frac{tp}{tp + fn} \tag{2.1}$$

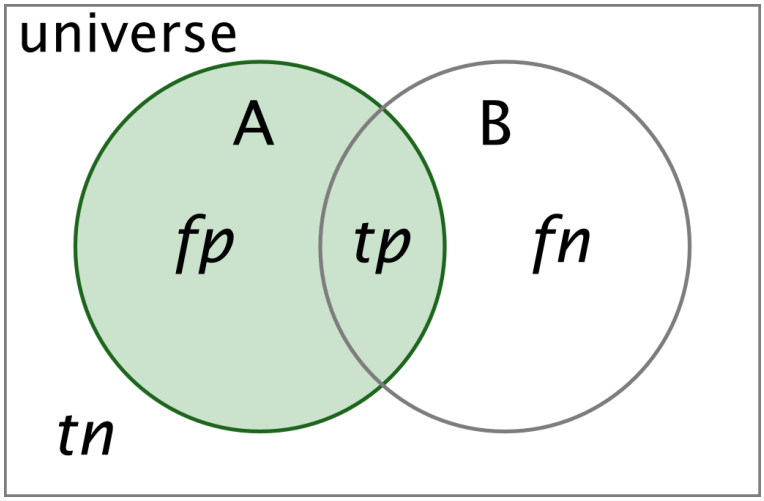


Figure 2.4: A figure demonstrating true positives (tp), false positives (fp), true negatives (tn), false negatives (fn). A is the set of data points labelled as positive, while B is the set of data points that are actually positive.

The general formula for the F-measure is

$$F = (1 + \beta)\frac{precision \cdot recall}{\beta^2 \cdot precision + recall} \tag{2.2}$$

The most commonly used version of this formula is the F_1 or the balanced F-measure:

$$F_1 = \frac{2 \cdot precision \cdot recall}{precision + recall} \tag{2.3}$$

which gives equal weighting to precision and recall.

The F-measure[7] is often used for NLP tasks such as named entity recognition or part of speech tagging, where each word does or does not belong to a certain class. Thus if a clear classification decision is made the F-measure is perfectly adequate. However, if using a method that gives a ranking of possibilities, then the F-measure is not suitable due to the non-discrete nature of the output. This is a common problem in the field of IR where the documents are ranked and the performance depends on where in the list of returned results the cutoff point is being made. In this thesis in particular, the probabilistic algorithms

[7]Throughout the thesis we will also alternatively refer to this value as the F-score or the F_1.

assign a score between 0 and 1 (a probability) for each label, which can be interpreted in several ways. The automatic response is to assign the label to the highest scoring category, in the binary case, this is the one with the probability over 0.50%. However, for imbalanced sets, where one class outnumbers the other, this is usually not where the optimal cutoff point lies and thus it has to be estimated in another way (Yang, 2001). That is, expecting the positive class to be chosen with a high probability will lead to high precision, while choosing a low probability will lead to higher recall. It is possible to approximate the cutoff point by holding back a portion of training data, however this leads to less training data for model estimation, or a further retraining with the full set of data.

In order to avoid estimating the cut-off point, a more reliable measure can be used. In IR this is usually accomplished by mapping how the precision changes as recall is increased, and an optimum point can be found in that way. In order to reduce a graph to a single value, the area under the precision-recall curve, or the *average precision* (AP), can be calculated for each query (Aslam et al., 2005). The AP is calculated so that for the top r results, each true positive result the inverse of the document rank is added together. This sum is then divided by the total number of positives. When evaluating a retrieval system, testing is performed over a series of queries on different topics. The *mean average precision* (MAP) measure gives the a succinct summary of the AP measures across the queries on a particular topic (Van Rijsbergen, 1979; Manning et al., 2008; Sanderson and Zobel, 2005). Unfortunately, it is not particularly obvious how the MAP measure should translate into the classification domain.

In testing classification algorithms there are two general strategies. Either a dataset is provided segmented into a training and test components, or this division is left up to the users. For evaluative studies, such as BioCreative (Hirschman et al., 2005a; Krallinger et al., 2008b), teams are given a sample of data for training in order to design their systems. Then the systems are evaluated against a new dataset, hitherto unseen by the designers. If the available dataset is very small, it is usually made available completely, and

the evaluation is performed on randomly selected unseen points. In general, the training points are randomised and the whole set is partitioned into n folds. The algorithm is trained on $n-1$ folds while one is reserved for testing. Traditionally, $n = 10$, and for best statistical analysis of the results, this 10-fold *cross-validation* experiment is performed ten times with different randomisations of the dataset (Kohavi, 1995).

One could consider each fold a query and calculate the MAP across a hundred randomised folds. This requires evaluation at different values of r, which give snapshots of algorithm performance at different levels. It was found, however, in Polajnar and Girolami (2009a) that this strategy leads to a large variance, and therefore a statistically indistinguishable set of values for the different classification algorithms. Manning et al. (2008, chap. 8) likewise mention that the MAP scores can vary across different queries.

For classifier comparison, the area under the receiver operator characteristic (ROC) is a more common measure than the MAP (Cortes and Mohri, 2004). The ROC is a plot of the *true positive rate* $\left(\frac{tp}{tp+fn}\right)$ (equivalent to recall) vs. the *false positive rate* $\left(\frac{fp}{tn+fp}\right)$. The closer the ROC is to the top left point of the space (0, 1), the larger the area under the curve (AUC), and the better the performance of the classifier. The AUC measures the probability that a positive test point is ranked higher than a negative one, thus if all of the positive test points are ranked higher than the negative ones the AUC is 1. The AUC is better than the F-measure for comparison of models that produce a ranking, because it does not assume a particular cut-off point for class participation. However, on highly imbalanced sets where there are many more true negatives, the ROC may be skewed (Manning et al., 2008; Davis and Goadrich, 2006, chap. 8). Therefore, for classification model comparison it is important to use a measure that shows trade off between the true positives and false positives, which can be done with either ROC or precision-recall curves; however, the relative performance of algorithms is preserved across both of the measures so they can be used interchangeably (Davis and Goadrich, 2006). In this thesis, the classification methods are being compared using the AUC.

In order to facilitate the comparison of the results, each result presented includes the

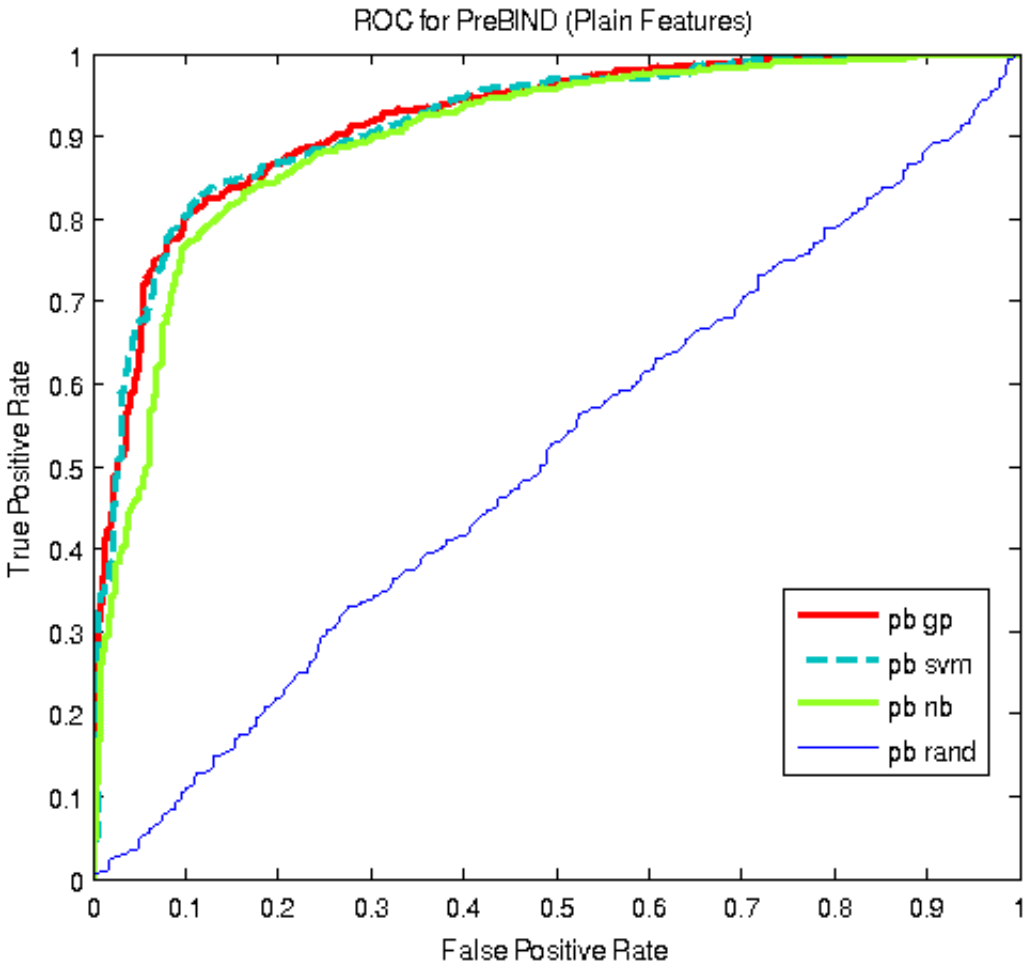


Figure 2.5: An example of an ROC comparison between three algorithms (Gaussian processes (gp), support vector machines (svm), and naïve Bayes (nb)) and random guessing on a single dataset.

standard error calculation. Significance tests are also applied to the important results in each of the experimental chapters. There are many significance tests, although Wilcoxon and Student's t-test are most popularly applied in the text retrieval and extraction context (Sanderson and Zobel, 2005). It was found that Wilcoxon confirms all of t-test results that demonstrate high certainty, while disagreeing with some borderline p-values. For example, in some cases where the t-test p-value is slightly larger than the usual 0.05 cut-off point, the Wilcoxon can indicate a rejection of the null hypothesis at the 5% significance. Therefore, borderline t-test results should be considered as contentious. Sanderson and Zobel (2005) experimentally demonstrate that, in the context of information retrieval, the

t-test produces lower error rates than Wilcoxon regardless of the underlying distribution of samples; hence, the results in this thesis are evaluated using the t-test.

2.3 Interaction detection methods

Automatic protein interaction detection can be useful in several different scenarios. There are, therefore, many different approaches for information extraction in the biomedical text, even just for the task of PPI detection. Some applications are geared towards helping with automatic population of interaction databases (Donaldson et al., 2003; Alex et al., 2008a), while others aim to support a wide variety of users by bridging the gap between the search engines and highly customised relation extraction software (Jenssen et al., 2001; Chen and Sharp, 2004; Rinaldi et al., 2007; Koster et al., 2006). In this section we examine different approaches to PPI detection roughly categorised into pattern-based, information retrieval-based (IR), and classification-based (Marcotte et al., 2001; Donaldson et al., 2003; Giuliano et al., 2006; Rosario and Hearst, 2005).

Pattern-based systems consist of hand-coded or automatically induced templates based on sample interaction sentences. The templates, which are sometimes scored for quality, are used to scan text and retrieve any matches. These patterns are usually unable to cover the wide variety of ways with which the interactions can be described in text. For this reason, these methods usually have high precision and lower recall. It is often offered as an argument that experimentally validated relations will be reported several times, thus affording more chance for the interaction to be retrieved (Thomas et al., 2000; Miyao et al., 2006). Conversely, this approach may only retrieve well known interactions, and as such not be very helpful to a researcher looking for novel interactions in a field that she is familiar with.

On the other side of the spectrum are the methods that consider any co-occurrence of two proteins in a sentence as a possible interaction. This assumption leads to a large number of retrieved interactions, unfortunately with a very low precision rate. A favourite

approach of initial systems aiming to construct interaction networks on the fly from user queries, it is an efficient way of allowing the user to browse potential interactions (Chen and Sharp, 2004; Jenssen et al., 2001; Rebholz-Schuhmann et al., 2007). More advanced IR-based approaches incorporate interaction detection into the indexing process (Rinaldi et al., 2007; Koster et al., 2006). This allows for fast retrieval of highly detailed information. However, for new types of interactions or entities to be included, the entire collection needs to be re-indexed. While it is possible to fully parse and index the collection of MEDLINE abstracts, this may not be a manageable solution for a large collection of full text articles. In addition, full parsing approaches rely on a pipeline of components that have to be adapted for the domain.

Finally, there are the (mainly supervised) classification-based methods (Cohen and Hersh, 2005; Marcotte et al., 2001; Donaldson et al., 2003; Katrenko and Adriaans, 2006; Bunescu et al., 2005; Erkan et al., 2007). These methods require samples of sentences that are at the very least annotated for relevance if not for the full interactions. On the other hand, they are fully automatic, apart from the labelling process. The availability of the standard data, such as AImed (Bunescu et al., 2005) and the LLL (Cussens and Nédellec, 2005), has allowed for faster development and testing of new algorithms, as well as for comparison across different approaches (Cohen and Hersh, 2005; Airola et al., 2008; Erkan et al., 2007; Pyysalo et al., 2008).

2.3.1 Pattern-based

Sekimizu et al. (1998) use a shallow parser to identify noun phrases that constitute subjects and objects of verbs frequently occurring in MEDLINE. They construct an algorithm to extract the interactions by locating head nouns of interaction indicating verbs such as *activates, binds, etc.* Similarly, Friedman et al. (2001) define a parser-based interaction extraction system, which forms a part of the larger project called GeneWays (Rzhetsky et al., 2004). It likewise performs noun phrase chunking and relation identification; however, it is informed with a hand-compiled knowledge base, which guides the generation

of a complex, nested, representation of the sentence meaning.

Thomas et al. (2000) adapt a tool designed to extract information in the finance news domain to do high-precision interaction extraction based on three verbs: *associates, binds, interacts*. After analysing 200 abstracts, they chose these specific verbs because they considered them most likely to show interaction between proteins as opposed to proteins and other entities. Their system scans unlabelled text, based on a set of patterns manually compiled from the sampled abstracts. The sentences that match the patters are then used to generate templates, which are scored for quality based on some pre-defined criteria. The templates can be used to find further interactions. There are several variations of the semi-automatic pattern creation algorithms. For example, SUISEKI (Blaschke and Valencia, 2001) likewise detects entity names and then matches the sentence fragments around the names against a predefined set of patterns. They find that precision of pattern matching reduces as the distance between the proteins grows. In general, these patterns are frames containing specific hand-derived grammatical constructs found around pairs of proteins. However, Hao et al. (2005) provide an automated method that learns patterns from a corpus of example interaction sentences. Their patterns consist of a combination of part of speech tags, where proteins have a unique tag, and restrictions created by sets of words that are likely to occur with those tags.

Handcrafted patterns are unlikely to cover the variety of ways interactions can be expressed in natural language. Bunescu et al. (2005) do a comparative study which shows that automatic pattern construction methods can provide higher precision. They also find that manually annotated protein names provide a better basis for pattern construction than the automatic ones. In constructing the PPI classifier from dependency-parser features, Katrenko and Adriaans (2006) examine the role that the interaction verbs contribute to the model. They find that given two proteins, the presence of a verb such as *activate* or *interact* in the verb phrase connecting the proteins is not enough to accurately predict a PPI. This indicates that limiting the vocabulary to look for specific clues reduces the coverage of pattern-based extractors.

2.3.2 Information retrieval-based: applications with a search engine

As search engines are still the primary resource for biomedical research, much of the current research concentrates on general applications that bridge the flexibility of information retrieval with the precision of information extraction. There are several implementations of these hybrid systems available for research purposes on the Internet. However these results were not comparable as they were evaluated on different data sets (Hersh, 2005).

In this section we outline two primary approaches that achieve this goal. The earliest systems assumed that entity co-occurrence was an indication of entity interaction, and that such evidence was sufficient in guiding researchers. More recent systems attempt to provide a deeper analysis of the data by using parsing.

2.3.2.1 Word co-occurrence models

Most of the word co-occurrence models use a local copy of MEDLINE, preprocessed for a set of named entities (Rebholz-Schuhmann et al., 2007; Jenssen et al., 2001; Hoffmann and Valencia, 2004). They also include a search component driven by user queries and a visualisation interface that allows for browsing of the term co-occurrences.

For example, EBImed (Rebholz-Schuhmann et al., 2007) is a search-based tool that integrates several popular biomedical text mining resources. It retrieves MEDLINE abstracts from a local collection based on an article ID or keyword query, and finds relevant terms from the Gene Ontology (GO) (Ashburner et al., 2000), UniProt (Wu et al., 2006), and lists of terms for drugs and species. The results are presented as a table of co-occurrences between the types of terms, with the more frequent co-occurrences rated higher.

Another tool, PubGene (Jenssen et al., 2001) produces a network of interactions based on protein/gene co-occurrence in a sentence. The entities are nodes in the graph, while the thickness of the edges represents the number of times the two entities are encountered together. Similarly, iHOP (Hoffmann and Valencia, 2004) remaps the entire MEDLINE as a navigable network of hyperlinks. The protein/gene named entities are links to all

the documents containing these entities. If there are synonyms, the website allows you to choose from a list of alternatives.

Chilibot (Chen and Sharp, 2004), on the other hand, assumes a different approach where the queries are lists of possible interactants specified by the user. The tool then creates pairwise queries, including any known synonyms for the given terms. These queries are then sent to PubMed and the retrieved abstracts are processed for term co-occurrences within the sentences. The results are presented, similarly to PubGene, as a network map of interactions; however, this network only contains the specified queried terms, while PubGene also includes any other encountered entities.

These approaches are difficult to evaluate comparatively as they have not been evaluated on a standard test set. There is no TREC-style query-based dataset that also has PPI annotations for the retrieved documents. However, Pyysalo et al. (2008) applied a co-occurrence test to standard PPI corpora and found high recall and a lower precision, consistent with IR methods, as well as the estimations reported for the above approaches. This effect is due to every co-occurrence being considered an interaction, a strategy that leads to a large number of false positives.

2.3.2.2 Models using preprocessing

The goal of more accurate PPI retrieval can be achieved by parsing the whole collection and subsequently querying for relations. Although ever expanding, the MEDLINE collection is still relatively small and grows at a slower pace than some other subsets of the Internet, for example, daily news. With increase in available computational power[8], it is possible to preprocess the entire collection, and then to index the daily updates as they become available. There are several systems that have exploited this fact to offer the flexibility of information retrieval and the power of relation extraction. They rely on storing information locally from a parsed version of MEDLINE in order to provide search engine access to the deeper knowledge in the abstracts. These approaches require a quality

[8]Both Miyao et al. (2006) and Koster et al. (2006) relied on multi-node clusters for parsing, while Rinaldi et al. (2007) only used a subset of MEDLINE

parser tuned especially for biomedical literature (Miyao and Tsujii, 2005; Koster et al., 2007; Clegg and Shepherd, 2007) .

For example, MEDIE (Miyao et al., 2006) use a parser to segment sentences into relevant phrases and use named entity recognition to find several classes of named entities, not only proteins or genes. The parsed corpus is then stored in a searchable format. The user queries are formulated in three fields (*i.e. subject, verb*, and *object*) which are then translated and matched against stored relations. The semantic search engine outlined in Rinaldi et al. (2007) and PHASAR (Koster et al., 2006, 2009) have similar architecture to MEDIE, although each uses a different parser, different data storage, and different query strategy. These approaches generally lead to higher precision and lower recall. This can be considered an advantage given the fact that many verified interactions will be reported several times in a large corpus (Miyao et al., 2006).

2.3.3 Classification of interactions

IR-based methods typically return all interactions between particular entities. On the other hand, it is possible to model a particular task and customise it using a training dataset and a supervised machine learning approach. If a researcher is interested in a particular type of relation, given a record of positive and negative samples previously encountered in their searches, one can build a model that will repeat the task. Although this is a highly personalised interpretation of this task, it is similar in essence to the following classification-based systems.

With the emergence of several standard datasets, there has been a proliferation of different approaches to classification-based protein interaction detection. In this section we describe a few of the relevant works. These can be roughly divided into *shallow-feature* and *deep-feature* methods. Shallow features are easy to extract from the text, requiring minimal preprocessing of the data. For example, the *bag-of-words model* (Lewis, 1998), consists of only the words contained in the document, usually excluding the most frequent words such as determiners and pronouns, often referred to as *stop words*. Other types of

shallow features include part of speech tags, named entities, and perhaps even sentence structure features resulting from a shallow parser. Deep features, in contrast, are generally the results of dependency parsing or full parsing of each of the sentences.

Marcotte et al. (2001), Donaldson et al. (2003), and Giuliano et al. (2006) all use shallow features in different ways. Marcotte et al. (2001) design a custom Bayesian model that discriminates between positive and negative abstracts based on the frequency of particular words that occur in them. Firstly, training data, consisting of abstracts that contain PPIs (positive) and ones that do not (negative), was sampled from a larger corpus of yeast-related MEDLINE citations. Given these abstracts, positive and negative discriminative words were chosen based on the deviation of their frequency in the training data from those occurring in the larger yeast corpus. To limit the overfitting of the model to the training data, the protein, gene, and pathway names were removed from the feature list. Using the 83 selected words, they constructed a model that calculates the probability that an abstract describes an interaction based on the frequencies with which these words occur in the positive and negative abstracts. Donaldson et al. (2003) also train on positive and negative abstracts (PreBIND), but instead of a Bayesian approach they use a support vector machine (SVM) to build a non-probabilistic discriminative model. The features are bag-of-words, however, only strings of letters are considered words, and these are cropped to the maximum length of 10 characters. In addition, only the top 1500 features with the highest information gain are used. Giuliano et al. (2006) likewise use the SVM and shallow features to detect interactions. Their approach is, never the less, very different as they are identifying pairs of interacting proteins in sentence corpora and aim to take advantage of the fact that the SVM is a kernel classifier (Section 3.2.2). First of all, they require corpora that are annotated for protein entities and interactions between pairs of entities. From the LLL and AImed corpora (Section 2.2.2), they extract all pairs of entities co-occurring within each of the sentences. The pairs that interact form the positive examples. They construct several different feature spaces based on the global and local contexts of the pairs. The global context feature spaces are bag-of-word representations of

the words occurring around the interacting pairs without excluding any stop words. The local context feature spaces contain descriptive properties of the candidate entities, such as the part of speech tags and the orthography. The feature spaces are all transformed into kernels and the learning is performed on the combination of these kernels. Thus, whilst they use shallow features, they are able to separate the sentence into segments relevant to each of the interacting pairs.

Katrenko and Adriaans (2006) use a similar approach of extracting training and testing examples from LLL and AImed, however they draw features from dependency parses of the sentences. The features that they consider are the first common ancestor of the two interactants and their children. They examine several different classifiers including naïve Bayes and find that an ensemble learning approach leads to the highest F-score. Bunescu et al. (2005), Erkan et al. (2007), and Airola et al. (2008) also consider dependency-parse features, but they use the resulting trees with with graph kernels and kernel classification algorithms such as K-nearest neighbours and SVM .

The above methods consider either regions of text or pairs of proteins, but they were all based on models that assume all possible types of interactions are relevant. In contrast, Rosario and Hearst (2005) formulate a different problem. Instead of considering a binary relevance problem, they classify interactions into several types, as derived from the HIV-1 Human Protein Interaction Database (Fu et al., 2009). In fact, starting from a given interaction from the database record, annotated with the two protein names and the ID of the supporting document, they find all the sentences in the article that describe the interaction. Using these sentences, they assign the interaction one of 10 classes. They present a dynamic graphical model, based on shallow features, that is simultaneously able to both identify the words in the sentence that perform the role of the interactants and also classify the sentence.

2.4 Discussion

Most recent research in PPI extraction concentrates on examining system performance in realistic scenarios. In particular this involves graduating from using the freely available MEDLINE abstracts to full text analysis. BioCreative II challenge Task II (Krallinger et al., 2008a) offers an overview of the state-of-the-art in several different aspects of full text PPI extraction for assisted curation.

The easiest challenge in Task II was the identification of articles that contain PPIs based on the abstracts. The highest performing submission on this task had the F-measure of 0.78 and the AUC of 0.86. This is an encouraging result as the full-text articles are expensive, while the abstracts are free. Significant cost-reductions in curation can be made if relevant articles are accurately located. This is task is slightly more difficult than the one that constitutes the initial part of the PreBIND system (Donaldson et al., 2003), which locates abstracts that contain PPIs with a cross-validated F-score of 0.90.

The next challenge was to extract the interacting protein pairs from the full articles, however with each of the interactants correctly linked to their SwissProt ID. This proved to be quite difficult with the best performing systems getting the F-measure of 0.35. Most systems that were evaluated on the AImed dataset have performance between F-measure of 0.60 and 0.70 in cross-validation (Bunescu et al., 2005; Katrenko and Adriaans, 2006; Erkan et al., 2007; Airola et al., 2008) . This is much higher, because these systems are compared against the hand-annotated interactions and do not aim to confirm the interactants with a protein database. In general, the expected performance of a relation extraction system is the product of the performance of the NER components for the involving entities and the component that identifies the interaction word. Cohen and Hersh (2005) and Hirschman et al. (2002) observe that, although, the protein named entity recognition systems have the F-measure in the 0.75 to 0.8 range, the initial relation extraction systems also had the F-measure in the similar range. This was likely the bi-product of the way that the systems were evaluated. For example, many of the initial systems assumed a set of interacting verbs that they would concentrate on, essentially ensuring that the perfor-

mance of the relation extraction depends solely the square of the performance of the NER system. Given the much harder task of extracting PPIs from the full-text articles, with normalised interactants the lower F-score is more realistic. The name normalisation was difficult because of difficulties in protein nomenclature including synonymy across different species. Thus the performance of the best system would be equivalent to the square of the performance of the normalising NER system multiplied by the interaction detection, whose F-score we can expect to be in the neighbourhood of 0.7, at best. In fact, the better performing systems had more sophisticated NER components (Krallinger et al., 2008a).

Better performance, (F-measure=0.65), was obtained on the third subtask of BioCreative II Task II, which consisted of locating the experimental description. However, the most interesting results manifested themselves in the fourth subtask, which consisted of finding the passages that best summarised the interaction description. It was shown that there exists a gap in the way biologists view the problem and the way the systems perform in the challenge. The annotators only chose one relevant passage, while the automatic systems tended to retrieve more, resulting in low precision and high recall.

One of the top performing systems (Alex et al., 2008c) in the BioCreative challenge was developed as an assisted curation tool and was tested in experiments in a real database population scenario (Alex et al., 2008a). They showed that there was about 1/3 improvement in the amount of time that was spent in extracting the interactions. Their evaluation demonstrated a preference for higher precision rather than higher recall. Unfortunately, they also found some reluctance on the behalf of curators to adopt and rely on the new technology. Perhaps more multi-purpose tools with a familiar search engine interface, such as MEDIE (Miyao et al., 2006) or PHASAR (Koster et al., 2006), could offer a bridge into further adoption of text mining solutions for biomedical research.

This chapter introduced protein interactions and an overview of different approaches to their detection. Included, alongside, were resources and measures that were required for comparison of methods, but which will also be used in the following chapters of the thesis.

Chapter 3

Supervised PPI Sentence Detection using Kernel Classifiers

Some of the most recent approaches to PPI extraction are pattern recognition methods based on classification. These classifiers are trained on collections of example sentences that describe protein interaction, which have been provided by biologists (Section 2.3). The process of modelling a task based on an annotated set of given examples is known as *supervised learning* (Kotsiantis, 2007). This can be contrasted with methods that try to infer categories from statistical patterns in the data without relying on labels, referred to as *unsupervised learning* (Bishop, 2006; Manning et al., 2008). These types of learning are just the opposite ends of a spectrum, in the middle of which we can find *semi-supervised learning* (Abney, 2007; Chapelle et al., 2006) approaches, which learn from a mix of labelled and unlabelled data. The rest of this chapter contains a brief introduction, followed by the descriptions of the input data and the chosen algorithms categorised by the three types of learning: supervised, unsupervised, and semi-supervised. The chapter concludes with a discussion.

3.1 Introduction

The PPI detection methodology described in this thesis is based on the model of assisted curation described by Donaldson et al. (2003), where a system can be built upon data with relatively simple relevance annotations, which can be created during the curation process. The aim is to detect the regions of text, in particular sentences, that describe protein interactions, rather than performing full interaction extraction. The reasoning behind this choice is that full interaction extraction yields a rather low F-score of around 0.65 (Bunescu et al., 2005; Katrenko and Adriaans, 2006; Erkan et al., 2007; Airola et al., 2008; Rosario and Hearst, 2005). On the other hand, by simplifying the problem to locating just an abstract describing a PPI, Donaldson et al. (2003) also significantly simplify the annotation task and simultaneously increase the F-measure to 0.90. Although efficiency calculation for assisted curation is complicated (Alex et al., 2008a), highlighting sentences that potentially contain PPIs could increase the overall speed of database population (Donaldson et al., 2003).

The simpler formulation of the task and the associated annotations also affects the data representation. In order to represent PPIs' constituent components (proteins and the interaction indicators), many of the classification-based approaches encode the dependencies between these terms as trees or graphs (Bunescu et al., 2005; Airola et al., 2008; Erkan et al., 2007; Kim et al., 2008). These are sometimes referred to as *deep features*. On the other hand, we are able to use simple *shallow features* (Giuliano et al., 2006), including the *bag-of-words* implementation (Lewis, 1998), which will be fully described in Section 3.2.

Many of the recently proposed classification-based approaches employ support vector machines (SVMs) (Section 3.3.1) to predict interactions (Donaldson et al., 2003; Bunescu et al., 2005; Airola et al., 2008; Giuliano et al., 2006), because in evaluations against other classification algorithms SVMs show the best performance (Erkan et al., 2007; Sugiyama et al., 2003). The SVM relies on geometrical discrimination between two classes, and while this does not necessarily influence the classification performance, it can cause dif-

ficulties in the training stage, the interpretation of results, and algorithm extension.

Once a geometric boundary is defined, the classification process is fairly simple: does a point belong to the positive or the negative side of the divided space? Defining the dividing hyperplane, unfortunately, is more difficult as in practise, data is often noisy and not linearly separable. The SVMs tackle this problem in two ways, firstly by transforming the data into a coordinate space where it may be separable (Section 3.2.2), and secondly by relaxing the training criteria. Ideally, the training process should produce a boundary which maximises the separation of the positive points from the negative ones; but if such a boundary is not feasible, the algorithm has a parameter that manages the trade off between enforcing the rigour of the separation, while still allowing some points to be on the wrong side. This parameter is almost always necessary and requires experimental tuning, leading to a substantial increase in training time.

The SVM produces a binary judgement, a test point is either positive or negative, so what is the difficulty in interpretation? In assisted curation, for example, it may be more advantageous to have a value associated with the confidence of group membership, i.e. how certain is the SVM that a point is positive or negative. While we can get the distance from the geometric boundary as classification output, this does not translate into an accurate representation of the probability of class membership (Figure 3.5 in Section 3.3.2).

Even if the output could be interpreted probabilistically, there is still a further drawback to the geometric framework: it is not as conducive to algorithm extensions as the fully probabilistic approach. The particular difficulty lies in efficient extensions for the multiclass problem, as is described in Section 3.3.2.

In order to address these issues, this thesis proposes the adoption of Gaussian process (GP) classifiers (Section 3.3.4), as the probabilistic alternative to the SVM. GPs are a Bayesian classification method analogous to the SVM that has rarely been applied to text classification; however, the probabilistic framework within which it is defined allows for elegant extensions that particularly suit text mining (TM) tasks (such as the sparse (Lawrence et al., 2003, 2005), semi-supervised (Rogers and Girolami, 2007), multiclass

(Rogers and Girolami, 2007), and multiexpert GPs (Rogers et al., 2010)). For this reason we seek to evaluate GPs and compare them to the more frequently used SVMs and Naïve Bayes (NB) (Section 3.3.3) (Lewis, 1998) classifiers. Both GPs and SVMs are non-parametric, meaning that they scale with the number of training documents, learn effectively from data with a large number of features, and allow for more relevant information to be captured by the data. The GP classifier, likewise, employs the same training data transformation as the SVM.

Nevertheless, while GPs have properties similar to SVMs (Rasmussen and Williams, 2006, pp. 141–146) they have failed to attract the same kind of attention in the text processing community. They have been applied to a variety of other bioinformatics tasks, such as protein fold prediction (Girolami and Zhong, 2007; Lama and Girolami, 2008) and biomarker discovery in microarray data (Chu et al., 2005). GPs have also been applied to text classification in a few instances. Online Gaussian Processes (Chai et al., 2002) and the sparse GP implementation, the Informative Vector Machine (IVM) (Stankovic et al., 2005), were investigated for multiple class text classification on the Reuters collection. Song et al. (2008) introduce sparse GP method, with lower memory requirements than the SVM, and apply it to the problem of automatic tag suggestion for social networking and bookmarking websites. In addition, GPs and SVMs were compared for preference learning on the OHSUMED corpus (Chu and Ghahramani, 2005b) and an extension of GPs for sequential data, such as named entities, was proposed by Altun et al. (2004).

Chapter 5 will experimentally compare these algorithms on several datasets and with different types of extracted features. From these results we will choose the best feature types and try to improve on their performance by introducing semantic information. This information is gathered by two different unsupervised word co-occurrence models (Chapter 4) and provides a smoothing of the features by re-weighting them based on their usage in the general biomedical literature. These unsupervised methods each give us different views of the data. In order to avoid having to use only one of these views we use an algorithm closely related to the GPs that allows us to combine different views of the data

into a single classifier (Section 3.3.5).

The supervised learning algorithms are described in Section 3.3, and can be contrasted with the descriptions of the unsupervised and semi-supervised approaches in Sections 3.4 and 3.5, respectively.

3.2 Training data and feature extraction

The data consists of biomedical text segmented into documents, where a document can be an article or a part of one, such as a sentence or an abstract. The labelled training data sets consist either of abstracts or sentences and are described in Section 2.2.2.

Each document can be viewed as a stream of characters that is then segmented into tokens. A token is defined by a regular expression, which is used to scan the strings and to extract all the matching candidates. In biomedical text, words are very different from ones that are seen in other domains (see Section 2.1.2 for some examples). They can contain numbers, hyphens, and apostrophes, as well as mixed capitalisation. Therefore, tokenisation decisions can drastically change the number of unique features that are found (Figure 3.1). Tokenisation is also the first step of many other language processing tasks, such as part of speech tagging and parsing. Due to such a different lexicon, all of these tools need to be customised for the biomedical domain.

In a *bag-of-words* (Lewis, 1998; Joachims, 1998) representation each unique word is considered a feature. A document is represented by the number of times each word occurs regardless of its order. Tokenisation and filtering are used to create a mapping from raw words into a reduced feature space.

In any given corpus of natural language there will be many words which appear often, but hold little discriminating information. This known as Zipf's law (Manning and Schütze, 1999, Chap.1), and these *stop words* are usually removed to cut down the size of the feature set. If we examine the GENIA corpus we can see that the top occurring words as in many corpora are pronouns, determiners, and conjunctions. One way to remove stop

F1	F2	F3	F4	F5
Mqo-negative	nls-gal3p	nls-gal3p	ytafs	mauretanicu
localizes	q-1	circuitri	eqplptpvtd	circuitri
ER-lumen	localizes	cyclin-independ	diffusely	orthogon
circuitry	seven-blad	q-1	diheterozy	macropinocyt
Nl-EST1	glc2	seven-blad	localizes	polypeptid
EDTA-inactivated	fluorocyti	glc2	polypeptid	interchang
localized	circuitry	mat-mc	interchang	deltaga
metacyclic	s-cdks	ygl257c	fluorocyti	op
tandem-repeats	localized	glc3	nrs	putamen
Swh3p	islet-spec	mitochondrial-loc	circuitry	crosstalk
Sac6	ygl257c	pex13p-contain	cancels	perineur
Gly58	gpi-anchor	sec18p-drive	saturating	enterokinas
# features: 47940	38304	35238	21050	16472

Figure 3.1: Examples of feature words that will be used in Chapter 5 and throughout the thesis. The different word processing techniques (from left to right) show an increase in feature abstraction. Feature type F1 is unnormalised. Feature types F2 and F4 limit words to length 10, while F3 and F5 employ stemming. Stemming results in condensing of several words that have the same root, into a single feature. In F1-F3 the words include dashes and numbers, but in F4-F5 the words are limited to sequences of letters. The total count of features (shown at the bottom) decreases as the tokens become shorter and represent more unique words.

words is to cut the highest frequency terms; however, closer examination shows that top ranked words also contain some biological words, such as *the, of, in, and, to, a,* ***cells****, that, by, with, is,* ***expression*** *...* Consequently, a list of commonly occurring non-biological English words is used to control the filtering process[1].

By normalising the words in different ways we can also combine semantically similar terms in a principled way. For example in Figure 3.1, in the first feature set (F1) we have unnormalised words, in the second we limit the word length to 10, and in third we apply *stemming*. Stemming is a process by which words are reduced to their root, so that *binding, binds*, and *bind* are all mapped to the same feature (Porter, 1997). In the last two feature examples (F4 and F5) all numbers and symbols are discarded. This leads to merging of some biomedical terms such as *glc2* and *glc3* into *glc*, whilst hyphenated words are considered as two separate features. Some terms, such as *q-1*, disappear as they get reduced to a single letter. In all of the feature sets, apart from F1, the words are also changed to lowercase, this combines words with different capitalisation into a single feature.

[1] ftp://ftp.cs.cornell.edu/pub/smart/english

3.2.1 Vector space representation of the input data

Next we can transform the data into a form that is suitable for classification by employing a feature mapping. Given a set of features that represents the training data, we map each element to an integer index. In that way each document, whether a sentence or an abstract, can be represented as a row vector in the data matrix $\mathbf{X} \in \mathcal{R}^{M \times N}$. Considering M documents containing N unique features $(w_1, \ldots, w_j, \ldots, w_N)$, the i^{th} document corresponds to the vector $\mathbf{x}_i = [x_{i1}, \ldots, x_{iN}]$ where each x_{ij} is a count of how many times the feature w_j occurs in the document i. When the features are words in a document, this representation does not preserve the word sequence or the semantics of the word associated with its placement relative to the other words in the document. $\mathbf{x}_i$ denote elements of $\mathbf{X}$, while $\mathbf{x}_*$ is a vector of a document that is being tested. The class labels, y_i, corresponding to each document, $\mathbf{x}_i$, are stored in the $M \times 1$ vector $\mathbf{y}$.

3.2.2 Transformation of the input space

The kernel function transforms the $M \times N$ input data to a square, positive semi-definite, $M \times M$ matrix, called the *kernel* (Cristianini et al., 2002). A matrix $\mathbf{K}$ is positive definite when for any vector $\alpha \in \mathcal{R}^M$, $\alpha^T \mathbf{K} \alpha \geqslant 0$. Kernel construction is governed by a set of closure properties (Cristianini et al., 2002, Chap. 3). The key closure property, which will be used in Chapter 6 for construction of semantic kernels is that a new kernel can be constructed by embedding a positive definite matrix into a linear product:

$$\kappa(\mathbf{x}, \mathbf{z}) = \mathbf{x}^T \mathbf{K} \mathbf{z}$$

The kernel matrix represents the similarity or distance between the training vectors, in a possibly infinite dimensional space. The way kernel the transformation is employed by classification algorithms will be described in Section 3.3.

The two kernel functions used in this thesis are the cosine:

$$\kappa_c(\mathbf{x}_i, \mathbf{x}_j) = \frac{\mathbf{x}_i \cdot \mathbf{x}_j}{|\mathbf{x}_i||\mathbf{x}_j|} \tag{3.1}$$

and the Gaussian:

$$\kappa_g(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\frac{|\mathbf{x}_i - \mathbf{x}_j|^2}{2\sigma^2}\right) \tag{3.2}$$

where the parameter $\theta = \sigma^2$ requires tuning. Cosine distance is commonly used to judge the distance of two documents (Manning and Schütze, 1999; Manning et al., 2008); while the Gaussian kernel, also sometimes called the radial basis function, or RBF kernel, has been found effective on biomedical text, namely, the PreBIND dataset (Donaldson et al., 2003), as well as other text data (Joachims, 1999).

3.3 Supervised algorithms

This section describes the classification algorithms used in this thesis. Classification is a supervised learning task which endeavours to place a new data point within one of a set of predefined classes. All of the algorithms use the same vector-based description of data, but have different interpretations of the space. SVMs treat the space as a geometric concept, where the classes can be separated by a hyperplane described in the N-dimensional space. The class of the new points is assessed based on the class of the points that are on the same side of the hyperplane. The NB, GP, and the pMKL algorithms, on the other hand, try to estimate the density of the training data with probabilistic distributions. A new point is assigned the label of the class to which it belongs with highest probability.

3.3.1 Support vector machines

The SVM (Vapnik, 1995; Joachims, 1998; Shawe-Taylor and Cristianini, 2004; Rasmussen and Williams, 2006) is a binary classifier that finds the optimal hyperplane separating the classes. The class labels are given as $+1$ for positive and -1 for negative

training data vectors, while the test points are assigned a class based on which side of the hyperplane they are located. This is judged according to a discriminant function $y_* = \text{sign}(\mathbf{w}^T\mathbf{x}_* + w_0)$, where $\mathbf{w}$ is the weight vector and w_0 is the offset. If all the data points lie on the correct side of the hyperplane, indicating that the data is linearly separable, then the following is true:

$$y_i(\mathbf{w}^T\mathbf{x}_i + w_0) \geqslant 1 \tag{3.3}$$

Because there may be many solutions that fit this constraint, the goal of training is to find the weight vector describing a hyperplane with the maximum distance from both positive and negative data points. The hyperplane is defined by the perpendicular projections of the vectors on the outskirts of the data, for which this Equation 3.3 equals 1. The support vectors are illustrated in Figure 3.2.

These *support* vectors are identified through an optimisation process that maximises the margin. Formulated as a constraint problem the defining vectors are marked by non-zero Lagrange multipliers, λ_i in the solution function:

$$y_* = \text{sign}(\mathbf{w}^T\mathbf{x}_* + w_0) = \text{sign}(\sum_{i=1}^{M} \lambda_i \mathbf{x}_i^T\mathbf{x}_* + w_0) \tag{3.4}$$

Figure 3.3.1 shows a two dimensional data set separated by a margin whose direction is defined by two of the negative and one of the positive points. From this figure it is clearly visible that if the data set was not linearly separable, such an optimal solution would be difficult to find.

3.3.1.1 SVMs for linearly non-separable data

There are two ways of improving the SVM for linearly non-separable data and they can be used in conjunction with each other. Firstly, by using the *kernel trick* (Aizerman et al., 1964; Boser et al., 1992), the inner product calculations in Equation 3.4 can be transformed to an alternative space ($\kappa(\mathbf{x}_i, \mathbf{x}_*)$) where the data may be linearly separa-

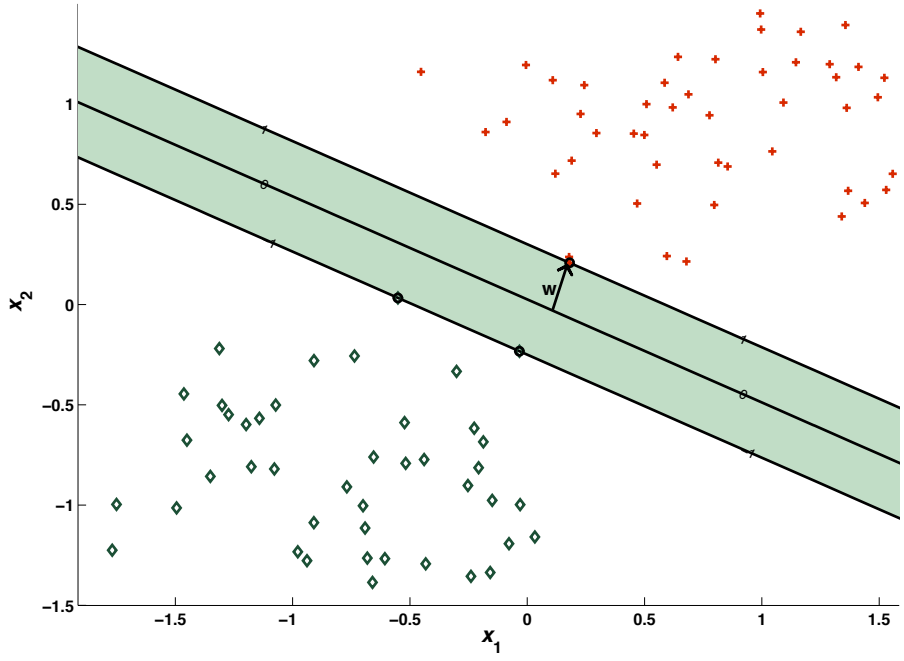


Figure 3.2: Example of a hard margin SVM for separable classes. The two dimensional data vectors $\mathbf{x}_i = [x_{i1}, x_{i2}]$ are plotted a plane. The negative ($\diamond$) and positive ($+$) training examples are separated by the middle line, while the surrounding shaded areas denote the margin. The middle line is the separating hyperplane, $\mathbf{w}^T\mathbf{x}_* + w_0 = 0$, while the top line is the positive margin $\mathbf{w}^T\mathbf{x}_* + w_0 = 1$, and the bottom line is the negative margin $\mathbf{w}^T\mathbf{x}_* + w_0 = -1$.

ble. Therefore instead of performing the above calculations on the data directly, we may perform them in the new space (Figure 3.3).

The second way of compensating for the noisiness of data relaxes the constraints that require all of the training data to lie on either side of the margin by allowing some cases where the constraint in Equation 3.3 can be violated. The amount of permitted transgression, during the training phase, is controlled by a margin parameter C. This parameter affects the minimisation of the weight vectors resulting in positive Lagrange multipliers not only for the support vectors, but also for points which lie beyond them, possibly on the other side of the separating hyperplane. This is called a *soft margin* SVM.

In practise most datasets contain some degree of noise, making the soft margin solution more appropriate; however, C has to be specified at training time, and the right choice

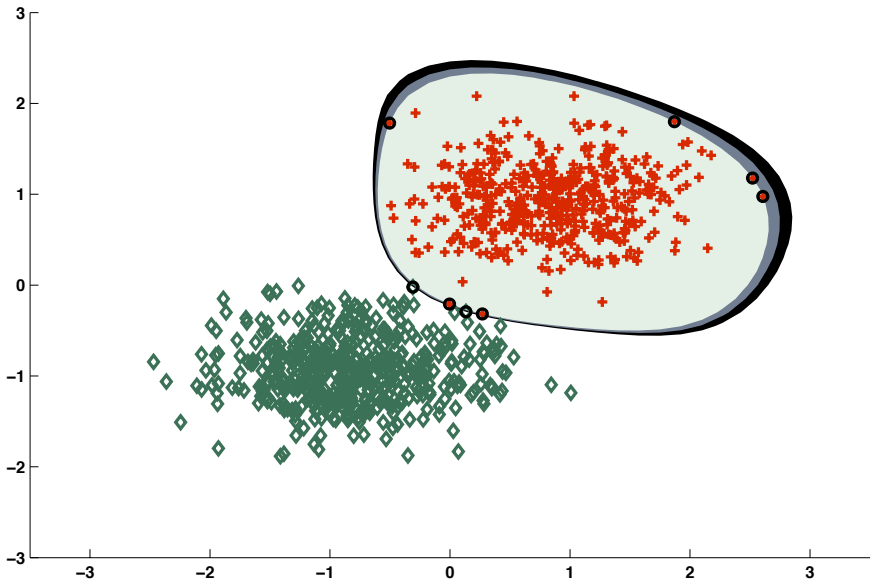


Figure 3.3: Example of a hard-margin SVM with an RBF kernel, for linearly non-separable classes. The negative ($\diamond$) and positive ($+$) training examples are separated by the middle line, while the surrounding shaded areas denote the margin. The axis of the graph represent the two dimensional data vectors $\mathbf{x}_i = [x_{i1}, x_{i2}]$.

has strong impact on the accuracy. As is shown in Figure 3.4, a lower value of C allows a larger number of training vectors to cross class boundaries. A high margin parameter corresponds to the original *hard margin* SVM.

An optimal value for C is generally obtained through cross-validation experiments and if a kernel with hyperparameters is used, these hyperparameters need to be adjusted along with C. As these values can vary from dataset to dataset, this tuning process can require a lot of computation before the final classifier is trained. For example, searching over a small grid of 10 possible values for the SVM parameter and 10 for one kernel parameter would require 100 separate cross-validation experiments. This is ten times as many experiments as the Gaussian process would need, because it only has kernel parameters.

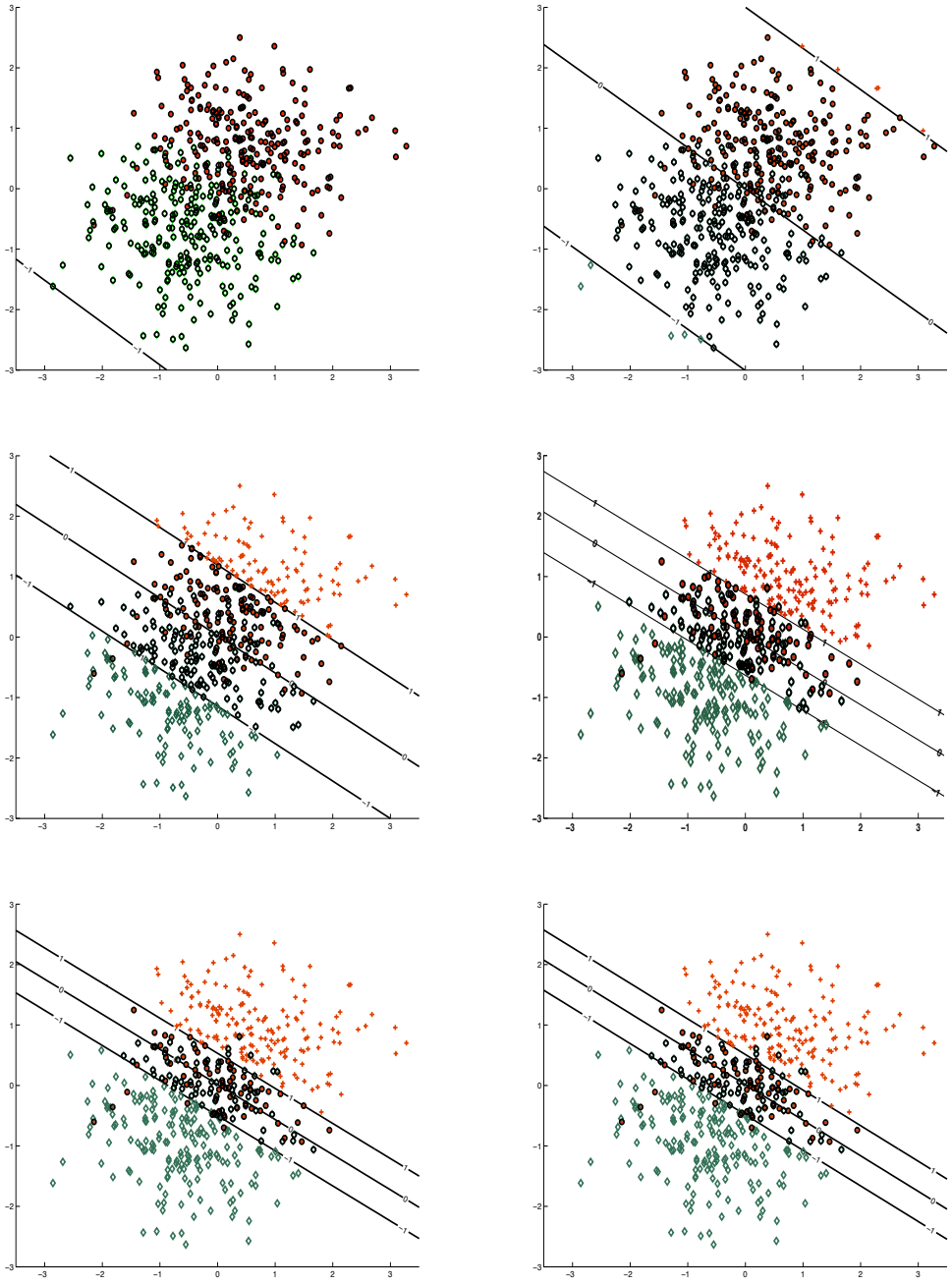


Figure 3.4: Tuning the margin parameter for the support vector machine, with values $C = \{0.0001, 0.001, 0.01, 0.1, 1, 10\}$. The axis of the graphs represent the two dimensional data vectors $\mathbf{x}_i = [x_{i1}, x_{i2}]$.

3.3.2 The multiclass and probabilistic extensions of the SVM

SVMs are a powerful tool that is applicable to a variety of classification tasks (Cristianini and Shawe-Taylor, 2000, Chap. 8), provided that the right kernel, kernel parameters and

margin parameter are chosen; however, the geometric framework of the algorithm makes extensions difficult.

For example, the SVM is essentially a binary classifier; and while there have been suggested improvements to handle multiple classes, most popular approaches involve combinations of binary SVMs, though theoretical multiclass extensions are available. Two popular combination strategies are *one vs. all* (OVA) and *one vs. one* (OVO) (Hsu and Lin, 2002). When using the former strategy one class is considered positive and the rest are negative resulting in K classifiers (where K is the number of classes), while in the latter approach each class is trained against each of the others resulting in $\frac{K \cdot (K-1)}{2}$ classifiers. In OVA classification a classifier is built for each of the classes and the training data is passed so that samples from each class, in turn, are considered positive, while the rest are considered negative. Each of the samples is assigned the class with the highest score. The OVO, on the other hand, is a polling method where a classifier is constructed for each pair of the classes. All of the samples are tested against all of the classes, and the class that gets the most votes is the one that gets assigned to the sample. Hsu and Lin (2002) examine the OVA and OVO methods as well as an improvement to the OVO method, which uses a directed acyclic graph (DAG) in the testing phase to resolve the best class assignment. They argue that OVA and DAG are the better than other combination strategies. Rifkin and Klautau (2004) show that OVA is an appropriate method, provided that the output from each of the classifiers is normalised in a way that allows accurate comparison between the predictions.

The multiple class SVM strategies are used often, for example, Ding and Dubchak (2001) use $\frac{27 \times 26}{2} = 351$ SVM classifiers, per feature space, to predict 27 protein fold classes. For the same problem, Damoulas and Girolami (2008) demonstrate how a single probabilistic multiclass kernel machine tailored to learn from multiple types of features for protein fold recognition can outperform a multiple classifier SVM solution. The proposed theoretical implementations are likewise computationally intensive. Lee et al. (2004) demonstrate the use of multiclass SVM on cancer microarray data that is

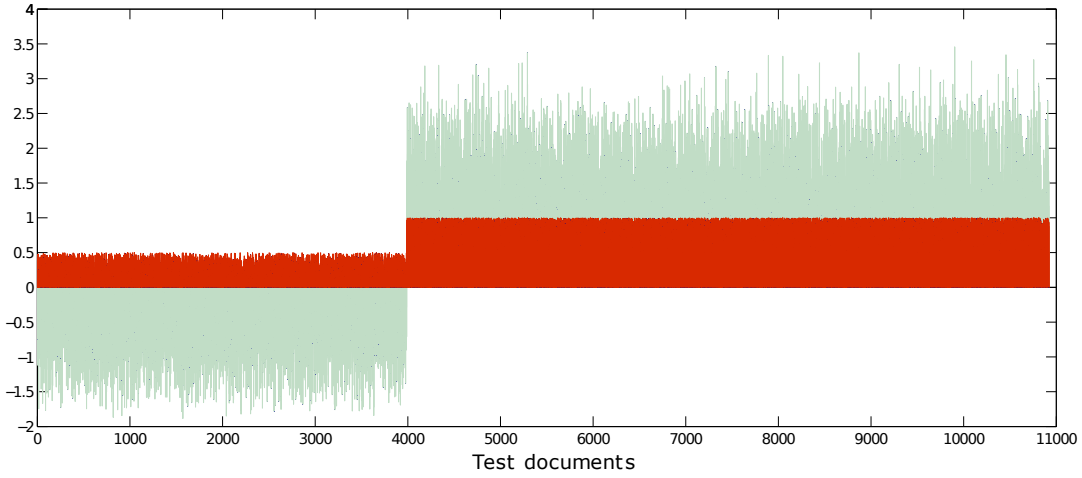


Figure 3.5: This figure shows the transformation of the SVM output through a probit function (inverse cumulative distribution function of the standard normal distribution $\mathcal{N}(0,1)$. The SVM output before the probit transformation is shown in green (lighter colour) and after in red (darker colour). The vertical axis represents the magnitude of the SVM output values and the probabilities simulated by the probit function. The horizontal axis represents ten different cross-validation cuts of the BioCreative data. The data examples are sorted by predicted label to show greater separation, thus the image is not a reflection of the classification accuracy.

$O(M^3K^3)$ (Crammer and Singer, 2001). More recently, Szedmak et al. (2006) demonstrated Maximum Margin Regression (MMR), a technique that can be adapted to multiclass classification; however, most applications still use combinations of multiple binary classifiers, as was the case in an application to the multiclass corpus of hierarchical relations in news text by Wang et al. (2006) and comprehensively presented in Hsu and Lin (2002).

The advantages of the probabilistic approach to classification have inspired attempts to develop probabilistic extensions of SVMs. For example, Platt (1999) proposed an *ad-hoc* mapping of SVM output into probabilities; however, this is not a true probabilistic solution as it yields probabilities that tend to be close together (Figure 3.5) (Rasmussen and Williams, 2006, p. 145). On the other hand, the GP and NB output probabilities give an accurate depiction of class membership that can be used to choose the optimal precision-recall trade off for a particular problem or further post-processing for appropriate decision making.

3.3.3 Naïve Bayes

The NB classifier is often used as a baseline for text classification problems (Rennie et al., 2003; Lewis, 1998). It generally gives good performance, although it is sensitive to the properties of data such as the feature dependence and class distribution (Rish, 2001; Yang, 2001; Rennie et al., 2003). In addition, as the number of features grows the predicted scores tend to diverge towards 1 and 0, and no longer provide true reflection of relevance for a class (Bennett, 2000).

The Naïve Bayes is a generative probabilistic classifier, and as such it does not determine a discriminative boundary like SVMs, but instead it is used to check whether a particular document was generated by a particular distribution (Nigam et al., 2006; Lewis, 1998). This classifier is the direct application of Bayes' rule:

$$p(c_k|\mathbf{x}_i) = \frac{p(c_k)p(\mathbf{x}_i|c_k)}{p(\mathbf{x}_i)} \tag{3.5}$$

This equation is interpreted as the probability of a class c_k given a document $\mathbf{x}_i$. Given a finite set of K classes we can assign the document to the most probable class. In order to compute the class probabilities we need to have values for all the components of the right-hand side of the equation. In particular, we wish to compute the likelihood of a document given a class, a value which can be estimated from training data.

The central (naïve) assumption is that the all the features are independent of each other, with respect to the class. So that:

$$p(\mathbf{x}_i|c_k) = \prod_{j=1}^{N} p(x_{ij}|c_k) \tag{3.6}$$

This assumption simplifies calculation of the likelihood, and even though it is clear that word order is essential for document understanding, it is possible to the determine the topic of a document from the words that occur in it most frequently. For this reason, the bag-of-words representation is often effective in document classification.

3.3.3.1 Multiclass NB

The experiments in this thesis use a multiclass NB that can be adapted to do semi-supervised learning via expectation-maximisation (Nigam et al., 2006). This is a mixture model, where each mixture component corresponds to a class. In this generative model, it is assumed that each document is created by drawing words from the distribution defined by the set of parameters $\theta = \{\theta_{c_k}, \theta_{w_j|c_k}\}$, which are learned from the training data. The probability of generating a document is calculated using the probability of its length, $p(|\mathbf{x}_i|)$ the probability of the feature occurring in the class, $\theta_{w_j|c_k} = p(w_j|c_k;\theta)$, and the probability that the class itself occurs, $\theta_{c_k} = p(c_k|\theta)$:

$$p(\mathbf{x}_i|\theta) = \sum_{k=1}^{K} p(c_k|\theta)p(\mathbf{x}_i|c_k;\theta) = p(|\mathbf{x}_i|)\frac{|\mathbf{x}_i|!}{\prod_{j=1}^{N} x_{ij}!}\sum_{k=1}^{K} p(c_k|\theta)\prod_{j=1}^{N} p(w_j|c_k;\theta)^{x_{ij}} \quad (3.7)$$

We use a Dirichlet prior on the parameters, with settings that lead to a Laplace smoothing (Nigam et al., 2006). This means that the estimates for the parameters $\hat{\theta}$ are calculated by adding one to the counts of the raw frequencies, in order to ensure that classes or features that do not occur do not lead to zero probabilities. Therefore,

$$p(c_k|\hat{\theta}) = \frac{1 + \sum_{x_i \in c_k} 1}{K + M} \quad (3.8)$$

and

$$p(w_j|c_k;\hat{\theta}) = \frac{1 + \sum_{\mathbf{x}_i \in c_k} x_{ij}}{N + \sum_{l=1}^{N}\sum_{\mathbf{x}_i \in c_k} x_{il}} \quad (3.9)$$

This is the most common type of smoothing, known as the *add one smoothing*, and while there are other methods, their effectiveness will depend on the data set and number of features used. He and Ding (2007) tested several smoothing algorithms on one data set. They vary the size of the training data and the number of features and found that given enough training data most smoothing methods have similar accuracies. When there are few training points and many features, more advanced smoothing methods perform better;

nevertheless, with under 5,000 features their results show that Laplace smoothing yields higher accuracy. It is difficult to generalise these results, because they are performed only on one data set; nonetheless, it can be concluded that marginal improvements in cross-validation accuracy can be obtained by choosing the optimal smoothing with the right number of features.

We can now use the above equations and Bayes' rule (Equation 3.5), to predict which class is most likely to have generated a new document $\mathbf{x}_*$, given the model that was learned from the training data:

$$p(c_k|\mathbf{x}_*;\hat{\theta}) = \frac{p(c_k|\hat{\theta})p(\mathbf{x}_*|c_k;\hat{\theta})}{p(\mathbf{x}_*|\hat{\theta})} \tag{3.10}$$

where $p(\mathbf{x}_*|\hat{\theta})$ is as defined in Equation 3.7.

3.3.4 Gaussian processes

The Gaussian process classifier is a discriminative probabilistic kernel method which, unlike the generative NB method, models the $p(y|\mathbf{x})$ directly instead of first modelling the document generation $p(\mathbf{x}|c_k)$. Figure 3.7 shows the probability landscape of a GP trained on two-dimensional data. The data is modelled using a latent function m, which is observed only at the evaluated points. This function is specified by the mean and covariance functions, $p(\mathbf{m}|\mathbf{X}) = \mathcal{N}(\mathbf{m}|\mathbf{0}, \mathbf{C})$, where the mean is $\mathbf{0}$, and the covariance $\mathbf{C} = \kappa(\mathbf{x}_i, \mathbf{x}_j) + \nu\mathbf{I}$. The kernel κ ensures that the values of m are close for the points that are similar. The choice of kernel and its parameters also regulate the smoothness of the function m. The addition of a small constant ν to the main diagonal of the kernel ensures against computational errors during inversion.

The function m can take any real number value, so in order to accurately model class probabilities, which are constrained to the range $[0, 1]$, the output is transformed using a function whose output falls within this range, *e.g.* a probit or a logistic function. Any choice of transformation function causes a non-conjugacy of the likelihood with the GP

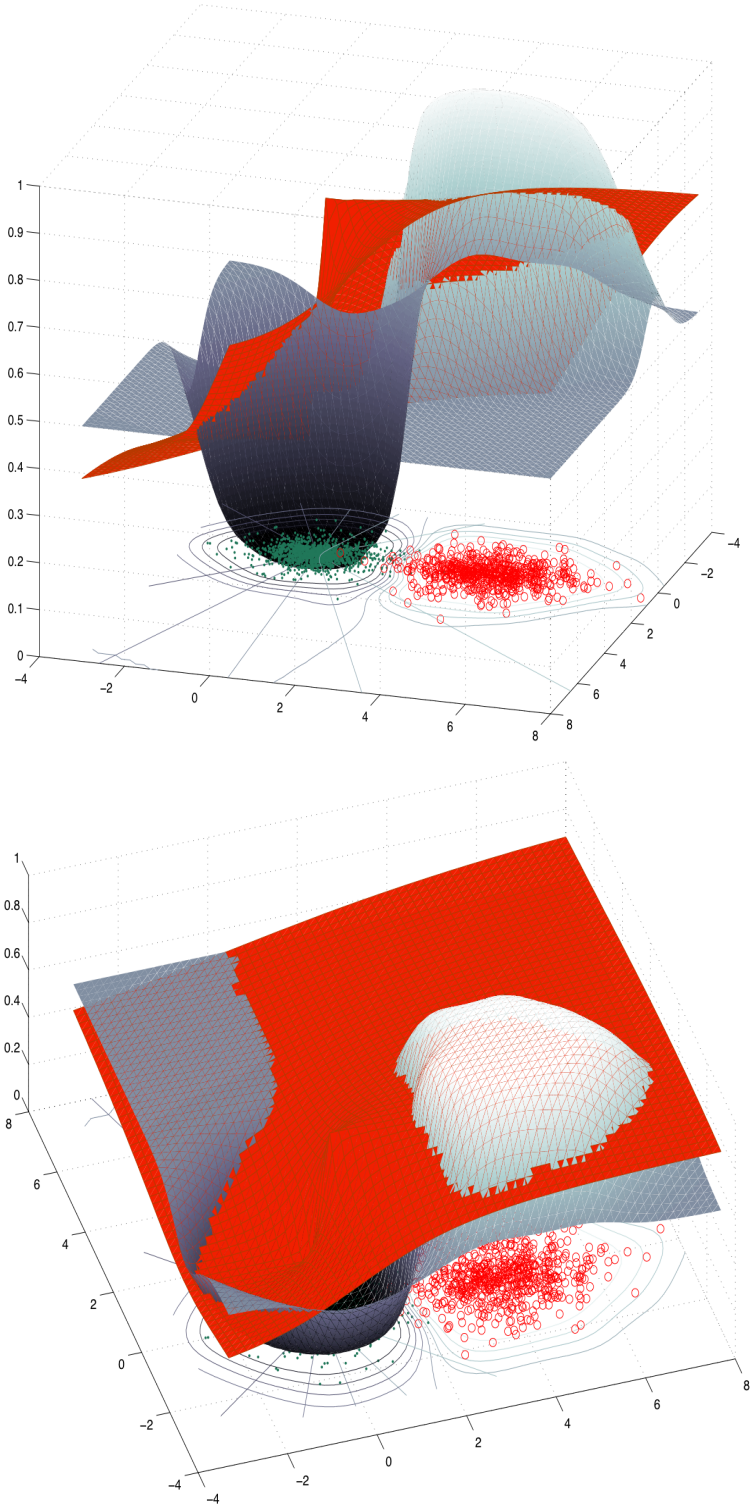


Figure 3.6: Two views of the probability landscapes returned by a GP trained with a cosine kernel (red) and a Gaussian kernel (grey). The x and y axes show the coordinates of the two dimensional data points, while the z-axis represents the value of the GP output probabilities.

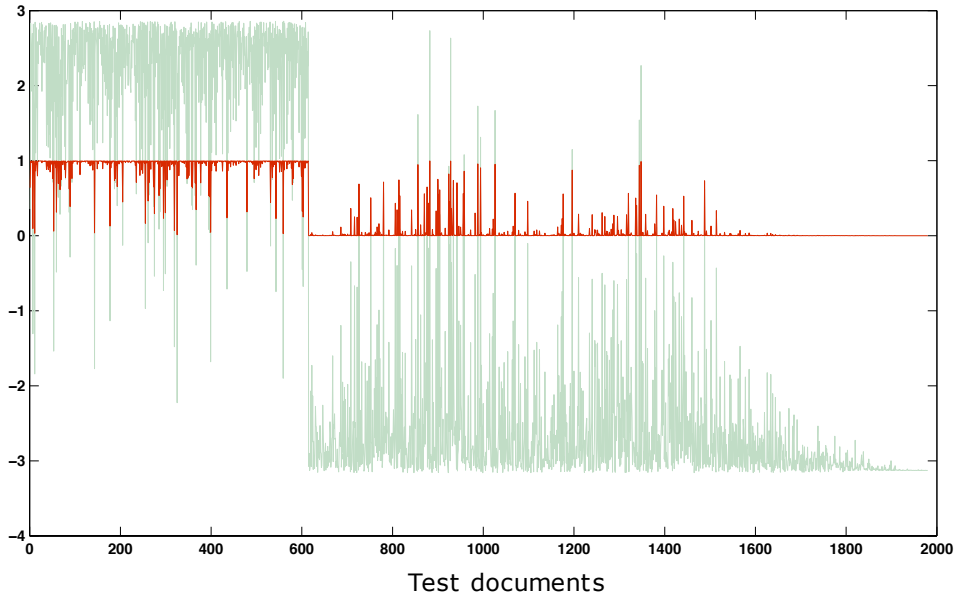


Figure 3.7: This figure demonstrates the GP likelihood before (green) and after (red) probit transformation. The y axis shows the value of the likelihood, while the x axis shows examples from the PreBIND dataset. Examples are sorted by true label.

prior, $p(\mathbf{m}|\mathbf{X})$, requiring either analytic approximations or sampling to compute the function posterior, $p(\mathbf{m}|\mathbf{X},\mathbf{Y}) \propto p(\mathbf{Y}|\mathbf{m})p(\mathbf{m}|\mathbf{X})$; however, this choice can influence the sampling strategy. We follow Girolami and Rogers (2006) and use the probit likelihood, $P(y_i = 1|m_i) = \Phi(m_i)$, which enables exact Gibbs sampling or efficient variational approximations through the auxiliary variable trick (Albert and Chib, 1993). This method allows us to reformulate the likelihood as an integral of a conditional probability of another variable, resulting in a form suitable for these particular approximations. Gibbs sampling is an example of Markov chain Monte Carlo (MCMC) family of stochastic sampling algorithms (Bishop, 2006, Chap. 11), which offer an exact solution given an unlimited amount of computation (Bishop, 2006, Chap. 10). Variational inference, on the other hand, is a deterministic approach that seeks to find a proposal distribution that best approximates the true posterior (Bishop, 2006; Albert and Chib, 1993, Chap. 11).

3.3.4.1 Multiclass and multiexpert GPs

The Bayesian framework allows for additional mathematical extensions of the basic algorithm, such as multiple classes (Rasmussen and Williams, 2006; Girolami and Rogers, 2006; Seeger and Jordan, 2004), sequential data (Altun et al., 2004), and ordinal classes (Chu and Ghahramani, 2005a). There are two extensions that are particularly useful for biomedical texts. This data is characterised by small annotated corpora gathered through expensive labelling processes from a large collection of freely available biomedical abstracts and Open Access articles.

Likewise, as was mentioned, the textual data is expensive to annotate. In order to ensure accurate annotation, most datasets are annotated by two or more annotators, at least partially. These sections are used to calculate inter-annotator agreement and create a uniform set of rules that would be used to consistently annotate the rest of the data. To deal with data in which a clear consensus may not exist, Rogers et al. (2010) have designed an extension for the GP that is able to learn from multiple annotators. Furthermore, the algorithm is able to give performance assessment scores to each of the annotators.

In a recent study, Wilbur et al. (2006) propose a set of five qualitative dimensions that can be used to annotate biological text, which, when used by 12 experts, result in a 70-80% inter-annotator agreement. This figure corresponds to a high level of disagreement and serves to show that whilst the choice and definition of the classes is important, so is the development of methods that incorporate diversity in opinion. Additionally, Cohen et al. (2005) emphasises that discussion of inter-annotator agreement may be crucial for wider usage of a corpus and the success of any prediction systems on which it is based. Although many corpora now include annotator disagreement statistics, few release the original annotations. Some differences in annotation can stem from valid ambiguities in the data, and the removal of conflicting annotations excludes potentially important information. This is true of many language related tasks; for example, Versley (2006) shows that in co-reference resolution some disagreement arises because a pronomial reference (*he, she, it, etc.*) does not always clearly refer to a single named entity.

The results in Rogers et al. (2010) indicate that knowledge encoded in the multiple annotations may be crucial for predictive systems. The multiexpert GP algorithm was inspired by the real dataset from the 2007 Computational Medicine Center (CMC) Medical NLP Challenge[2].

The original data consisted of anonymised medical records from a childrens hospital consisting of two parts, the medical impression and medical history. The medical history briey describes patients prior complaints, while the impression describes the results of the current examination. Each record also includes ICD9 codes[3] assigned by three different companies. The diversity of labels assigned by each company showed that the problem contains inherent ambiguity.

The dataset also has a single consensus or majority label assigned for each document, which was chosen by a disambiguation process from all the given labels; however, the results using a multiexpert GP show that training on the labels from all three experts improves on learning simply from the majority label. In addition, using the majority label is equivalent to using only one of the experts, while the other two are individually worse predictors of the majority label, they contribute when all three are used.

This work is not included in the main contributions of this thesis because while the relevant PPI data exists, it is not publicly available. Nevertheless, in their publication Alex et al. (2008b) acknowledge the possible relevance of the algorithms like the multiexpert GP:

> Multiply annotated documents were left in the corpus and not reconciled to produce a single, gold standard version. It was found during piloting that reconciliation could be very time-consuming so we decided to focus our resources on obtaining a larger sample of papers.

The multiexpert GP algorithm could be used not only to learn from the multiple annotators, but also to resolve disagreements in labelling.

[2]Data and detailed description on the CMC website http://www.computationalmedicine.org/challenge/index.php.
[3]http://icd9cm.chrisendres.com/index.php

3.3.5 Probabilistic multiple kernel learning (pMKL)

In Chapter 6, different views of data are generated by projecting word similarities onto the training data. In Chapter 7, these different views are combined in order to improve classification performance by taking advantage of all the information contained therein. A kernel-based algorithm that is similar to SVMs and GPs, but which can learn from multiple feature spaces is used for this purpose. Multiple kernel learning is used to translate multiple feature spaces into kernels, which are then combined with a particular weighting into a single composite kernel. This is in contrast to standard ensemble classification, where a separate classifier is used for each new feature space and the consensus labelling is then created from the different outputs.

Introduced by Damoulas and Girolami (2008), probabilistic multiple kernel learning (pMKL) is a probabilistic kernel machine that follows a generalised linear model structure. In Chapter 7 we employ the variational Bayes inference approach, denoted as VBpMKL, which follows GPs in employing the same likelihood and auxiliary variable trick. The model assigns the class labels y_i based on model parameters $\mathbf{W} \in \mathcal{R}^{M \times K}$ and the auxiliary variable z_{jk}, where M is the number of documents, K is the number of classes and k is a specific class, as before. The class label is assigned to the class k corresponding to the largest value in the $K \times 1$ vector $\mathbf{z}_j$. VBpMKL is similar to the non-linear GP classifier, except that instead of modelling via the latent function $\mathbf{m}$ it models via an $M \times 1$ row vector $\mathbf{w}_k$ which indicates the weight with which a training point $\mathbf{x}_m$ votes for class k.

We use two approaches for setting the kernel combination weight vector $\boldsymbol{\beta}$. In the first, the convex linear estimation approach, the kernel combination weights, β_s, for the $s = 1, \ldots, S$ kernels, are learned through sampling according to Equation 3.11. The final weights reflect the discriminative abilities of the kernels and their contributions to increasing the predictive likelihood of the model. This does not necessarily fully correlate with the accuracy of the model, which can be estimated only through validation or testing. The kernel parameters $\theta^{(s)}$ are fixed for each of the semantic kernels, and found through

cross-validation experiments.

$$k\left(\mathbf{x}_i, \mathbf{x}_j, \boldsymbol{\beta}, \boldsymbol{\Theta}\right) = \sum_{s=1}^{S} \beta_s k_s \left(\mathbf{x}_i^{(s)}, \mathbf{x}_j^{(s)}, \boldsymbol{\theta}^{(s)}\right) \text{ with } \sum_{s=1}^{S} \beta_s = 1 \text{ and } \beta_s \geqslant 0 \; \forall \; s \qquad (3.11)$$

In the second approach, we assume that the weights are fixed to $\beta_s = 1$ for each of the kernels. This manner of fixing the contributions of each of the kernels was found to produce slightly better results than enforcing the restriction that $\sum_s \beta_s = 1$, *i.e.* $\beta_s = \frac{1}{s}$. It leads to an unnormalised kernel where the diagonal is S and not 1.

Due to model similarity, VBpMKL is expected to provide similar results to the GP, while also offering the ability to do kernel combinations.

3.4 Unsupervised learning

Unlike the supervised approaches described above, unsupervised learning methods do not use labels to learn the structure. Instead they attempt to induce a structure directly from the data.

The main application is clustering, which is used to discover data groupings by observing the similarities between the points, as defined by some distance measure, such as the Euclidean distance (Manning et al., 2008, Chap. 16). Clustering algorithms can either give hard assignments of one cluster per document or soft assignments of multiple clusters per document. The K-means (Bishop, 2006, Chapt. 9) algorithm is the classic example of a single class per document clustering, it is based on geometric assignment of classes based on the centre of mass of a group of data points. K-means can be generalised using the Bayesian framework and the expectation-maximisation (EM) algorithm (Bishop, 2006; Manning et al., 2008). Both of these algorithms, however, require the number of clusters to be pre-specified. Hierarchical clustering methods, on the other hand, output a tree structure that details the proximity of documents. The documents themselves are leaves, and the root is a single class which combines them all. The middle children reflect

different levels of similarity between the documents (Manning et al., 2008, Chap. 17).

Topic models, such as latent semantic analysis (LSA) (Landauer et al., 1998; Papadimitriou et al., 2000) or latent Dirichlet allocation (LDA) (Blei et al., 2003) are an example of soft cluster assignment (Manning et al., 2008, Chap. 18). In a vector space representation of documents, topic models can be used to reduce the dimensionality of training data by converting a document-feature matrix into a document-topic matrix. At the same time, these topics can be interpreted as soft category assignments for the documents. LSA and LDA have been used on a variety of text-based linguistic tasks, such as synonym detection and automated essay grading (Blei et al., 2006; Zheng et al., 2006; Papadimitriou et al., 2000; Kakkonen et al., 2005) (as well as for other applications (Yuan et al., 2005)).

LSA and principal component analysis (PCA) (Gorban et al., 2007) are linear algebra-based techniques whose core component is the singular value decomposition (SVD) (Manning et al., 2008, Chap. 18) of the document-feature matrix. SVD separates the data matrix into three components $\mathbf{X} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T$. The diagonal matrix $\mathbf{\Sigma}$ contains the singular values in descending order. A reduced rank representation of the data matrix can be produced by setting all except the top d rows of $\mathbf{\Sigma}$ to zero and reconstructing $\mathbf{X}_d$, an $M \times d$ version of $\mathbf{X}$, such that $\mathbf{X}_d = \mathbf{U}\mathbf{\Sigma}_d\mathbf{V}^T$. It has been shown that this new representation of the vector space has properties that re-introduce the semantic links between words, which had been lost through the simplified bag-of-words document description (Landauer et al., 1998). For example, if two documents have no words in common their cosine distance would be 0; but if many of the words that they do share often co-occur together elsewhere, these documents would be closer in the reduced space.

Probabilistic latent semantic analysis (pLSA) (Hofmann, 1999) offers a generative model as a probabilistic interpretation of the LSA algorithm. pLSA, like NB, models each word in a document as being generated by a component in a mixture model. These components, which correspond to the classes in NB, represent the topics and the number of topics needs to be specified, as in LSA. A document is represented as a mixture of weights, which are derived based on the topics that the words in the document belong

to; unfortunately, these weights are not probabilities guided by a generative process like the word and topic distributions (Hofmann, 1999; Blei et al., 2003). Thus, the mixture weights are impossible to estimate for any new documents.

In the LDA model, the word order independence (with respect to the documents) assumption is maintained and a further assumption is made that the order of documents is independent with respect to the collection. Therefore, Blei et al. (2003) introduce another parameter that models the distribution of topics across a document, producing a fully generative model. The main advantage of this model is that it can now be used for prediction as well as document analysis, and any new documents outside of the training collection can be assigned probabilities with which they belong to specific topics. This approach also reduces the overfitting that is commonly associated with the pLSA model. In addition, the pLSA computational complexity scales with the number of documents, while the LDA model scales with the number of topics (Blei et al., 2003).

Chapter 4 describes in depth two more unsupervised methods Hyperspace Analogue to Language (HAL) (Lund and Burgess, 1996) and Bound Encoding of the Aggregate Language Environment (BEAGLE) (Jones and Mewhort, 2007). These semantic models are slightly different from the topic models described above. Whereas, in LDA and LSA words are generally grouped based on their co-occurrence in similar documents, in HAL and BEAGLE words are grouped based on their co-occurrence with other words. Like LSA, HAL and BEAGLE have been evaluated on a variety of psycho-linguistic tasks such as TOEFL word synonym examinations and semantic priming (Lund and Burgess, 1996; Jones et al., 2006; Jones and Mewhort, 2007; Landauer et al., 1998). In addition, while HAL preserves the entire dimensionality of the feature space, BEAGLE offers reduction through *random indexing* (Section 4.4.1), a technique that was also investigated as a faster alternative to SVD for LSA by Papadimitriou et al. (2000).

Even though in unsupervised learning we have unlabelled data, it is possible to gain insight into the structure of the data. For textual data, the separation into topics is a particularly good analogy and allows the categorisation flexibility that is inherent in the

linguistic medium.

3.5 Semi-supervised learning

On one side we have the costly labelled data that enables us to model a problem in a way that corresponds to the way people view it. On the other side, in the case of text, we have a multitude of data generated by humans, but not evaluated for particular tasks. Unsupervised learning gives us the ability to find certain structures in the data, but this segmentation into clusters or topics may have many different interpretations due to complexity of some problems. In between these two extremes, we have semi-supervised learning (SSL) (Chapelle et al., 2006; Abney, 2007), which leverages the labelled data with large amounts of unlabelled data in order to improve classification performance.

In traditional SSL, the shortage in labelled data is usually addressed by adding samples without class labels directly to the training set (Erkan et al., 2007). This approach generally leads to the greatest improvements in classification performance when there are few labelled sentences and many unlabelled sentences; however, semi-supervised learning is also volatile, and could lead to a significant loss in accuracy (Rogers and Girolami, 2007). There are several properties data must have in order to be suitable for this type of SSL, in particular, it must conform to the smoothness and cluster assumptions (Chapelle et al., 2006, Chap. 1). The *smoothness assumption* dictates that any two points in high-density areas of the data space must belong to the same class. The *cluster assumption* is related and requires the points in the same cluster to be in the same class, or rather that the decision boundary be in the low density areas. The implication is that the labelled and unlabelled data come from the same distribution and that the unlabelled data adds density to the areas already occupied by the labelled points. Unfortunately, gathering the data from the same distribution is not always easy. For example, if we examine the PPI datasets (Section 2.2.2), we can see that many of them are produced through a multiple step querying process that significantly changes not only the content, but also the

distribution of positive versus negative documents.

This thesis explores a different notion of SSL. Instead of including the unlabelled data directly into the training set, specific information about word usage is extracted from a large unlabelled corpus and integrated into the kernel space of the GP and VBpMKL classifiers. In this way we can exploit freely available biomedical texts regardless of the distribution of PPIs within them.

Chapter 6 contains an example experiment with traditional NB and GP SSL and a range of experiments with the enhanced kernels.

As mentioned before the advantage of the Gaussian process classifier is the Bayesian framework which allows for fluid extensions, including SSL (Rogers and Girolami, 2007)). SVMs can also be used for SSL (Silva et al., 2007); however, the GPs also permit you to combine semi-supervised and multiclass learning in a single classifier, following the semi-supervised extension to the multinomial probit classifier in Rogers and Girolami (2007). Essentially, the null category likelihood of Lawrence and Jordan (2006) is extended to the multi-class setting, by augmenting the problem with an additional *null* class, inside which no data (labelled or unlabelled) can exist. This has the effect of forcing the GP decision boundaries to lie in areas of low data density, thereby enforcing the cluster assumption (Lawrence and Jordan, 2006).

For the NB, SSL translates into a two step process, called expectation-maximisation (EM) (Bishop, 2006, Chap. 9) where the estimated parameters of a model are used to classify unlabelled data, which is then incorporated into the training data to give the new estimate of the model parameters (Nigam et al., 2006). This process is repeated iteratively while the log likelihood of the model is improving. The initial parameters are learned from the labelled data as described above.

The goal is to iteratively find the best estimate for the model parameters θ given the labelled ($\mathbf{X}_l$) and unlabelled training data ($\mathbf{X}_u$), $\hat{\theta} = \mathrm{argmax}_\theta p(\theta|\mathbf{X}, \mathbf{Y})$; where the $\mathbf{X}$ is the combined data, $\mathbf{X} = \mathbf{X}_l \cup \mathbf{X}_u$, and $\mathbf{Y}$ are the class labels. In the first, or the E-step, we estimate the expected probabilities of the class labels for the unlabelled data, based on

the old model parameters as per Equation 3.10. In the second step, called the M-step, we calculate the new estimate of the model, $\hat{\theta}$, by calculating the log likelihood

$$\ln(p(\theta|\mathbf{X},\mathbf{Y})) = \ln(p(\theta)) + \ln(p(\mathbf{X}_u|\theta)) + \ln(p(\mathbf{X}_l|\theta)) \tag{3.12}$$

where $p(\theta)$ is the prior knowledge from the labelled training data, $\prod_{k=1}^{K} \theta_{c_k} \prod_{j=1}^{N} \theta_{w_j|c_k}$, and

$$p(\mathbf{X}_u|\theta) = \prod_{\mathbf{x}_i \in \mathbf{X}_u} \sum_{k=1}^{K} p(c_k|\theta) p(\mathbf{x}_i|c_k;\theta) \tag{3.13}$$

For the labelled training data, we know the class labels y_i:

$$p(\mathbf{X}_l|\theta) = \prod_{\mathbf{x}_i \in \mathbf{X}_l} \sum_{k=1}^{K} p(c_k = y_i|\theta) p(\mathbf{x}_i|c_k = y_i;\theta) \tag{3.14}$$

By iterating through the E and M steps while the log likelihood is increasing we can find a best local estimate of the parameters (a local maximum). Multiple randomised initialisations of the algorithm can be used to find an estimate of the global maximum. In this approach it is important that the data is well separated into clusters and that the iterative increases in the log likelihood correspond to a decrease in classification error (Nigam et al., 2006).

3.6 Discussion

This chapter describes the different types of learning used in this thesis and the algorithms that will be applied. Supervised learning is the foundation of the PPI detection method used in this thesis. Therefore following the literature, we examine the use of the state of the art SVMs on several PPI datasets. In a series of experiments we compare the SVMs to the GPs. The GPs are a kernel method like the SVMs, but are also fully probabilistic and require less tuning, making them better candidates for exploration of semantic kernels. A related algorithm, VBpMKL, is used to combine several semantic kernels into a single

classifier. Finally, the baseline for these experiments is provided by NB.

The complexity of GPs, VBpmKL, and SVMs grows with the number of documents, while the complexity of NB grows with the number of features. When using the bag-of-words (Lewis, 1998) approach, the number of features grows with the number of documents. The sizes of the quality PPI data sets do not pose problems for these algorithms; however, for larger data sets it may be necessary to perform feature selection or dimensionality reduction when using NB, while for GPs and SVMs there are sparse implementations that learn only from a subset of the training examples. Most current SVMs are sparse, while the informative vector machine (IVM) is the sparse implementation of the GP. Apart from the IVM, the Bayesian framework of the GP lends itself to different extensions, including the semi-supervised, and multi-expert GPs presented in this chapter.

SVMs have benefited from widely available implementations, for example the C implementation SVM^{light} (Joachims, 1999), whose algorithm uses only a subset of the training data; however, informative vector machines (IVMs)[4] (Lawrence et al., 2005; Girolami and Rogers, 2006), which are derived from GPs, now offer an analogous probabilistic alternative. A naïve implementation of SVM has a computational complexity $O(N^3)$, due to the quadratic programming optimisation. Fortunately, with engineering techniques this can be reduced to $O(N^2)$, or even more optimally, to $O(ND^2)$ where D is a much smaller set of carefully chosen training vectors (Keerthi et al., 2006). Likewise, the GP has $O(N^3)$ complexity; with techniques such as the IVM this can be reduced to the worst case performance of $O(ND^2)$. On the datasets presented in this thesis, the difference for combined training and classification user time for GPs and SVMs, was similar despite the difference in implementation. The GP and VBpMKL were implemented in MATLAB while the SVM was implemented in C, and thus are not directly comparable. Using 64-bit cluster technology and optimised libraries ensured less than a minute difference in performance between the experiments on ten-fold cross-validation. SVMs did, however, slow down significantly if non-sparse matrices or matrices with small real values were passed

[4]http://www.cs.man.ac.uk/ neill/gpsoftware.html

to the algorithm. This is most likely due to implementation optimisation techniques.

Unsupervised learning provides the ability to induce a structure from data based without relevance judgements. It can also be used to reduce the dimensionality of the space. In this thesis PCA is used to represent data in a way that allows for easier visualisation, while LDA is applied to word co-occurrence matrices in order to examine word similarity by topic. Two further unsupervised algorithms described in Chapter 4 are used to generate these word-word matrices. These are in turn used to introduce semantic information into the GP classifier kernels, producing a novel approach to semi-supervised learning (Chapter 6). These semantic kernels have different parameters, whose exploration is enabled by the fact that GPs do not have an extra parameter that requires tuning, like C in the soft-margin SVMs. In addition, by choosing to use only one kernel at the time, the different views offered by each of the semantic models are not being used to their full potential. This hypothesis is explored in Chapter 7 using the VBpMKL algorithm.

Chapter 4

Biomedical Word Similarity Through Semantic Models

In natural language there are many subtleties of expression. Two words may be listed as synonyms in a thesaurus, but in general they are interchangeable only depending on the context in which they appear. The words are loaded with meaning associated by their cultural interpretation. These subtle changes in word interpretation are referred to as the *semantics* of the words (Gärdenfors, 2004; Eikmeyer and Rieser, 1981). The word semantics are exemplified by the synonymy and polysemy: the similarity of words to each other, and variations of a single word depending on the other words appearing around it. A classical example is the word *bank*, when it is followed by the word *robber* it refers to a completely different concept then when it is preceded by the word *river*. It is possible that both meanings of the word could occur together in close proximity, for example in the sentence: *The bank robber made a quick getaway by sliding down the river bank*.

When sentences are represented in the bag-of-word notation, as described in Section 3.2, the sequence of the words in the sentence is lost, and consequently, some of the clues pointing to their connotation. In addition, these subtle interpretations of word meanings are learned through exposure to speech and text from childhood (Riordan and Jones, 2007; McMurray, 2007); so it is unlikely that the small PPI corpora (Section 2.2.2) could

encode enough statistical information about the usage of the various words occurring within them. For this reason, this thesis explores the use of lexical semantic models, as a way to gather information about word meanings from large amounts of biomedical text. Two models, which are considered here, gather information about words based on their co-occurrence with their closest neighbours. These models are Hyperspace Analogue to Language (HAL) (Lund and Burgess, 1996) and Bound Encoding of the Aggregate Language Environment (BEAGLE) (Jones et al., 2006).

This chapter gives a brief introduction to semantic spaces; followed by the explanation of the vector space representation of words, which closely mirrors the document representations, described in Chapter 3, and by the descriptions of HAL and BEAGLE. The chapter concludes with a comparative discussion.

4.1 Introduction

Lexical semantic models were created as a representation of word meaning in vector space. Semantic models were initially conceived as a series of axes that represent the concepts embodied by a word (Osgood et al., 1957). These could be qualities such as size, *e.g.* a house is larger than a mouse. The words would be placed along these axes by human assessors (Osgood et al., 1957; Lund and Burgess, 1996). The reliance on hundreds of judgements prompted research into models that minimise the need for human input (Lund and Burgess, 1996). Word co-occurrence models, such as HAL (Lund and Burgess, 1996; Burgess and Lund, 1997; Burgess et al., 1998; Burgess and Conley, 1998; Song and Bruza, 2001; Rohde et al., 2005) and BEAGLE (Jones et al., 2006; Jones and Mewhort, 2007), represent a word in a space where the dimensions are represented by all of the unique words in a training corpus. These dimensions are referred to as *basis*, while the words placed as points in this space are called *targets* (Lowe, 2001). In HAL the placement of the target word in the space depends on their co-occurrence with the basis words within a specified distance in the original text. The more a target and a basis

co-occur the higher the magnitude of that basis vector, pulling the target further along that particular dimension. The semantic distance between the target word vectors is usually judged by a Cartesian distance measure such as cosine (Section 3.2.2).

The concept of basis is not limited to words, and as we saw in Section 3.4. Models such as LSA (Landauer et al., 1998) use documents as the basis; but many other mappings are feasible, for example Padó and Lapata (2007) describe syntax-based models which use portions of parsed sentences as the basis.

Depending on the representation, the number of basis can be quite large, thus some models seek to reduce them in a way that does not harm the performance. For example, in LSA the dimensionality is reduced by using SVD to combine the documents into topics (Landauer et al., 1998). SVD is computationally intensive, so alternative approaches have been considered. For example, Papadimitriou et al. (2000) propose a faster way to reduce the dimensionality of the document-feature space by using *random mapping* (Section 4.4.1, *i.e.* Kaski (1998)). BEAGLE uses random mapping to reduce the dimensionality of a word-word co-occurrence matrix; while Rohde et al. (2005) show that in their particular set of experiments removing rare basis words from the HAL model does not impede performance. In fact, using top 14,000 most frequent words is as effective as using all 100,000 basis.

The evaluation of both the lexical semantic models and the latent models, such as LSA (Landauer et al., 1998), pLSA (Hofmann, 1999), and LDA (Blei et al., 2003), is usually performed on a set of psycholinguistic tasks. The initial evaluation is usually carried out by observing the semantic neighbourhoods of words, either by printing the lists of similar words given a particular target, or by reducing the space down to two dimensions and observing the relative distances of the words on a single plane (Lund and Burgess, 1996; Burgess et al., 1998; Blei et al., 2003; Landauer et al., 1998). A popular evaluation is carried out on datasets containing pairs of words evaluated for their semantic similarity (Rohde et al., 2005; Jones et al., 2006; Landauer et al., 1998; Padó and Lapata, 2007). Others include assessment as part of a specific task, such as query expansion in

Information Retrieval (Azzopardi et al., 2005; Song and Bruza, 2001) or automatic essay grading (Kakkonen et al., 2005).

In this thesis, HAL and BEAGLE are applied to large biomedical corpora. Their ability to group words semantically is evaluated in the context of the PPI classification task.

4.2 Vector space representation of words

Statistical semantic models are generally represented in the vector space. Each word corresponds to a vector whose dimensions are called the *basis*. In general, there exists a mapping between the contexts and the basis. In a lexical model, if this mapping is 1-to-1, then the length of the target vectors is the number of all possible unique words occurring in the contexts, which may be equivalent to the number of targets. Therefore a target vector is $1 \times |B|$ vector of frequency counts, where B is the set of basis elements. There are $|T|$ such vectors, one for each target word in the set T. A non-zero entry in a vector represents the number of times the target coincides with a context word (basis) within the corpus. These counts can be transformed by a function of this frequency, such as *tf-idf* (Manning et al., 2008, Chap. 6). Figure 4.1 shows an example of a simple lexical semantic model in vector space built from a small two-sentence corpus.

A word-based model can result in high-dimensional space corresponding to the number of unique words in the corpus. To limit the dimensionality and remove some noise, highly frequent function words, also known as *stop words*, are usually ignored. These are usually excluded by removing a standard list containing most commonly occurring words including pronouns, determiners, and conjunctions. If the model does not take into account word order, function words contribute very little information.

Given an example corpus:

```
A cat sat on the mat.
The dog chased the cat.
```

The co-occurrence matrix is:

Targets \ Basis	cat	sat	mat	dog	chased
cat	0	1	1	1	1
sat	1	0	1	0	0
mat	1	1	0	0	0
dog	1	0	0	0	1
chased	1	0	0	1	0

Figure 4.1: An example of the vector space representation in a word-based semantic space, where the context consists of all the words co-occurring with the target within the sentence. The columns represent the basis words that make up the contexts, while the rows are the target words. In this model the co-occurrence matrix is symmetric. The stop words (*a, the, on*) are ignored.

4.3 Hyperspace Analogue to Language

Hyperspace Analogue to Language is a semantic model that represents word similarity according to co-occurrence within a window of specific length (Lund and Burgess, 1996; Burgess and Lund, 1997; Burgess et al., 1998; Song and Bruza, 2001; Rohde et al., 2005). The strength of word co-occurrence is determined by the distance between the two words within the specified window. This has the effect of boosting the similarity between words whose close contexts are the same, while allowing for variation in the phrasing of the context.

The HAL matrix, $\mathbf{H}_o$, is constructed by passing a window of fixed length, L, across the corpus. The last word in the window is considered the target and the preceding words are the basis. Because the window slides across the corpus uniformly, the basis words are previous targets. Therefore the set of targets T is equivalent to the set of basis B, $T = B$, and thus, the HAL matrix, $\mathbf{H}_o$, has the dimensions of $|T| \times |T|$.

The strength of the co-occurrence between a target and the basis depends on the distance between the two words, l, $1 \leqslant l \leqslant L$, within the window. The co-occurrence scoring formula, $L - l + 1$, assigns lower significance to words that are further apart. The overall co-occurrence of a target-basis pair is the sum of the scores assigned every time they coincide within the sliding window, across the whole corpus.

Even though the matrix is square, it is not symmetric. In fact the transpose of the

Given an example corpus:

```
A cat sat on the mat.
The dog chased the cat.
```

Ignoring the stopwords:

```
cat sat mat dog chased cat
```

The window of length $L = 5$ and the target word cat:

cat sat mat dog chased t:cat

The matrix $\mathbf{H}_o$ constructed by passing the window across text:

	cat	sat	mat	dog	chased
cat	1	5	4	3	2
sat	2	0	5	4	3
mat	3	0	0	5	4
dog	4	0	0	0	5
chased	5	0	0	0	0

The matrix $\mathbf{H}_o + \mathbf{H}_o^T$:

	cat	sat	mat	dog	chased
cat	2	7	7	7	7
sat	7	0	5	4	3
mat	7	5	0	5	4
dog	7	4	5	0	5
chased	7	3	4	5	0

Figure 4.2: Construction of a HAL matrix from a small two-sentence corpus with the window of length L=5. The stop words (*a, the, on*) are ignored.

matrix reflects the co-occurrence scores with the basis that occur within the window of length L *after* the target. Thus $\mathbf{H}_o$ and $\mathbf{H}_o^T$ together reflect the full context (of length $2L-1$) surrounding a target. There are two ways of combining this information so that it would be considered when the distance between targets is calculated. The first way is to concatenate $\mathbf{H}_o$ and $\mathbf{H}_o^T$ to produce a $|T| \times 2|B|$ matrix. The second way is to add the two matrices together $\mathbf{H}_o + \mathbf{H}_o^T$. We found that for our kernel combination method that the latter strategy is more effective. This was also the case when HAL was employed for query expansion (Song and Bruza, 2001). Therefore, from now on when we refer to a HAL matrix we will assume $\mathbf{H} = \mathbf{H}_o + \mathbf{H}_o^T$.

4.3.1 Probabilistic Hyperspace Analogue to Language

Azzopardi et al. (2005) propose an interpretation of the HAL, where the co-occurrence frequencies are transformed into a probabilistic estimate.

The probability that we encounter a target word t_i given the basis word b_j is the sum of the probabilities of t_i and b_j co-occurring at distance l:

$$p(t_i|b_j) = \sum_{l=1}^{L} p(l)p(t_i|b_j, l) \tag{4.1}$$

We can view pHAL as a decoupling of the original frequency-based HAL matrix into

separate matrices for each distance l. In other words, we can have L different matrices each one storing the co-occurrence frequencies of the targets and basis at l, $1 \leqslant l \leqslant L$. The $(i,j)^{th}$ entry of a matrix for a given distance l would contain the number of times that t_i and b_j occur at that distance, $||(t,l,b)||$. Then we can see that

$$p(t_i|b_j,l) = \frac{||(t_i,l,b_j)||}{\sum_{k=1}^{B}||(t,l,b_k)||} \tag{4.2}$$

which can be interpreted as a row-normalisation of each of the l matrices. We refer to these normalised matrices as $\mathbf{pHAL}_l$. The prior $p(l)$ is a scaling value determining the contribution of each of the $\mathbf{pHAL}_l$ to the final combined $\mathbf{pHAL}$ matrix:

$$\mathbf{pHAL} = \sum_{l=1}^{L} p(l)\mathbf{pHAL}_l \tag{4.3}$$

In HAL, the combination is a linear function $L-l+1$. The probabilistic interpretation of this is decaying function is $p(l) = \frac{L-l+1}{2\sum_{l=1}^{L} L-l+1}$, while it is also possible to have other priors such as the uniform $p(l) = \frac{1}{2L}$. Azzopardi et al. (2005) found that given $L = 5$, for purposes of query expansion in information retrieval, the uniform prior performs better than the linear prior.

4.4 Bound Encoding of the Aggregate Language Environment

The Bound Encoding of the Aggregate Language Environment (BEAGLE) model (Jones et al., 2006; Jones and Mewhort, 2007) was proposed as a combined semantic space that incorporates word context, $\mathbf{C}$, and word order, $\mathbf{O}$, encodings. It is constructed in the following way:

- In essence, BEAGLE context encoding is just like the simplified model in Figure 4.1, where the matrix contains co-occurrence frequencies and the context consists of words occurring in the same sentence as the target. As such, initially, the set of targets and basis words is the same, and both consist of all unique words in

the corpus.

- Where BEAGLE differs is in the representation of the frequency counts. The data is stored in a vector space reduced by random mapping, described below in Section 4.4.1. If a context word appears frequently in the same sentence as a target word, its signal will be amplified through addition. Words sharing the same contexts will have strong signals corresponding to the common words.

- BEAGLE is also able to record the order of the words through n-gram frequency. This is done in a separate matrix and the full BEAGLE model is the addition of the context and order matrices.

This section discusses the BEAGLE model construction by first describing random mapping, then the context encoding, and finally the order encoding.

4.4.1 Random mapping for dimensionality reduction

Random mapping, sometimes also referred to as random projection or random indexing, is a method for reducing the dimensionality of data. For large data matrices, methods based on matrix decomposition such as principal component analysis (PCA) or singular value decomposition (SVD) can lead to heavy computational overheads (Papadimitriou et al., 2000; Bingham and Mannila, 2001; Fradkin and Madigan, 2003). On the other hand, random mapping provides a computationally efficient method of dimensionality reduction with minimal distortion in the distances between vectors (Bingham and Mannila, 2001). It has been used for classification and clustering in a variety of applications including image and text (Bingham and Mannila, 2001; Kaski, 1998), software quality (Jin and Bie, 2006), databases (Achlioptas, 2001), and others (Fradkin and Madigan, 2003).

The mapping transforms an $|T| \times |B|$ matrix, $\mathbf{B}$, into a lower dimensional space by multiplication with the transpose of the $|B| \times D$ matrix of random values, $\mathbf{R}$. $\mathbf{R}$ can be constructed by random sampling from any distribution with the mean 0. The normalised rows form a near-orthogonal set of basis. The more dimensions are preserved, the more

orthogonal the vectors are. In other words, the matrix $\mathbf{R}\mathbf{R}^T = \mathbf{I} + \epsilon$, where ϵ is a small amount of noise that decreases as D increases (Kaski, 1998).

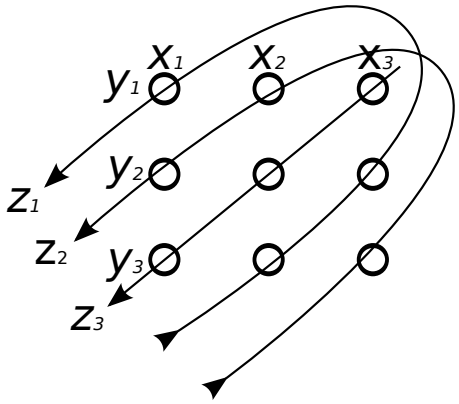
Random mapping is used in BEAGLE in order to decouple the word vector lengths from the size of the vocabulary, as well as to reduce the vector length in order to allow for more efficient execution of costly matrix operations that are needed to encode word order (Section 4.4.3).

4.4.2 Context encoding

The BEAGLE context matrix, $\mathbf{C}$, can be constructed by first building the $|T| \times |B|$ dimensional matrix of co-occurrence frequencies. This matrix shows how many times a target word t_i occurs within the same sentence as a basis word b_j, within the whole corpus. An example of such a matrix is shown in in Figure 4.1. This frequency matrix is represented in a reduced dimensional space. The *offline* way of reducing this space is by multiplying it by the transpose of a $|B| \times D$ matrix of random values, $\mathbf{R}$, (as described in Section 4.4.1). Alternatively, this procedure can be performed *online*, by generating the reduced representation sequentially as the corpus is traversed. The latter method is more advantageous in that it allows for an expandable lexicon and it eliminates the need to store and transform the large frequency matrix. Addition of new words through corpus expansion only requires addition of new rows to the matrix.

The number of dimensions D is chosen so that it is large enough to ensure that this vector is unique for each target or basis word. Jones et al. (2006) suggest that multiples of 1024 are an appropriate choice for D, and use $D = 2048$ to encode larger corpora.

In the online method, each unique word in the corpus is assigned a D-dimensional vector of normally distributed random values drawn from the Gaussian distribution $\mathcal{N}(0, \sigma^2)$, where $\sigma = \frac{1}{\sqrt{D}}$. The choice of the standard deviation of $\frac{1}{\sqrt{D}}$ ensures normalised vector lengths. These are referred to as environmental vectors, which are just rows of the random matrix $\mathbf{R}$ denoted by $\mathbf{r}_b$, where $b \in B$ is a basis word. The $|T| \times D$ BEAGLE matrix, $\mathbf{B}$, where the rows are indexed by target words, is initialised to 0. The corpus of sentences



$$\mathbf{z} = \mathbf{x} \circledast \mathbf{y}$$

$$\mathbf{z}_i = \sum_{j=0}^{n-1} \mathbf{x}_{j \bmod n} \cdot \mathbf{y}_{(i-j) \bmod n}$$

Figure 4.3: The left part of the figure demonstrates circular binding operation ($\circledast$) used to create word-order vectors in BEAGLE. The D-dimensional word environmental vectors $\mathbf{x}$ and $\mathbf{y}$ are combined to create the vector $\mathbf{z}$ that is likewise D-dimensional.

S is scanned in order, and for each target word t_i encountered, the context vector $\mathbf{c}_{t_i}$ is updated. Initially this context vector is empty. When scanning the text, $\mathbf{c}_{t_i}$ of an encountered target word t_i is updated by adding the sum of the the environmental vectors of the basis words, b_j, in this sentence. If we are only considering the contexts, the matrix entry for the target word t_i is the sum of the context vectors gathered from all the sentences s_k such that t_i occurs in s_k,

$$\mathbf{b}_{t_i} = \mathbf{c}_{t_i} = \sum_{s_k \in S} \mathbf{c}_{t_i \in s_k},\ \mathbf{c}_{t_i \in s_k} = \sum_{b_j \in s_k} \mathbf{r}_{b_j}$$

.

4.4.3 Word order encoding

The BEAGLE model also supports encoding of word order by employing a *binding* operation based on directional circular convolution (Figure 4.3) (Jones and Mewhort, 2007). The convolution operation ($\circledast$) compresses the cross-product matrix of two vectors to produce a vector of the same dimensions as the original operands, *i.e.* if we have two random vectors of dimensions $1 \times D$ their convolution, $\mathbf{r_i} \circledast \mathbf{r_j}$, also has dimensions $1 \times D$.

The word-order calculation is based on addition of n-gram bindings for each target word (Manning and Schütze, 1999, Chap. 6). An n-gram is a series of n sequential words in a sentence. A bigram contains the target word and either the preceding or the following word. As n grows, so does the number of possible combinations involving the target word and the $n-1$ surrounding words (as is demonstrated in Figure 4.3). In a binding operation, a target word t_i is represented by a placeholder vector $\mathbf{\Phi}$, sampled for the Gaussian distribution in the same manner as the environmental vectors. Meanwhile, the surrounding basis words are represented by their environmental vectors.

The order vector $\mathbf{o}_{t_i}$ (for a single sentence s_k) is represented as a sum over the bindings of all the n-grams of length $n \leqslant \lambda$:

$$\mathbf{o}_{t_i \in s_k} = \sum_{l=1}^{p\lambda-(p^2-p)-1} bind_{i,l} \tag{4.4}$$

where p represents the position of the word t_i in the sentence. The number of n-grams that can be constructed for a target depends on its position in the sentence. A target at the start of the sentence will have less possible constructions because all n-grams will start with the target itself, while more centrally-placed target will generate a larger number of n-grams. For example, for word *sat* in the sentence *A cat sat on the mat.* we have the following n-grams, $\lambda = 3$:

$$\text{Bigrams} : \begin{cases} bind_{sat,1} = \mathbf{r}_{cat} \circledast \mathbf{\Phi} \\ bind_{sat,2} = \mathbf{\Phi} \circledast \mathbf{r}_{on} \end{cases}$$

$$\text{Trigrams} : \begin{cases} bind_{sat,3} = \mathbf{r}_{a} \circledast \mathbf{r}_{cat} \circledast \mathbf{\Phi} \\ bind_{sat,4} = \mathbf{r}_{cat} \circledast \mathbf{\Phi} \circledast \mathbf{r}_{on} \\ bind_{sat,5} = \mathbf{\Phi} \circledast \mathbf{r}_{on} \circledast \mathbf{r}_{the} \end{cases}$$

In a BEAGLE matrix that considers both word context and order the vector for each target t_i is the sum of both of these views of each of the sentences s_k, in the corpus S, that

contain the target, $\mathbf{b}_{t_i} = \sum_{s_k \in S} \mathbf{c}_{t_i \in s_k} + \mathbf{o}_{t_i \in s_k}$.

4.5 Discussion

HAL and BEAGLE, the two models described in this chapter, provide two different views of word similarity based on co-occurrence within a large corpus. HAL is based on a sliding window of a specified length, and implicitly encodes the word order by weighting closer co-occurrences more strongly. BEAGLE, on the other hand, considers the context of a word to consist of all the words within the same sentence. For both models the semantic distance between the words depends on the frequency with which two targets occur with same basis. While for HAL this is visibly reflected in the frequency matrix; for BEAGLE the signature peaks of the random vectors associated with highly-frequent basis words become more prominent, making the overall fingerprints of the context vectors that contain similar basis closer together.

In the context of PPI classification, the models are interchangeable, they both provide information about the relationships between words. These relationships can be judged by distance functions, such as the kernel functions used in classification. The BEAGLE representation is more efficient as the word similarity data is compressed into a smaller number of dimensions. In addition, the only parameter that can be varied in the BEAGLE context encoding is the D, the number of dimensions; on the other hand, for HAL, it is necessary to examine the effect of the changing window size L, has on the performance of the classifiers.

BEAGLE, however, also supports a separate, optional encoding of the word order by tracking frequency of word co-occurrence within n-grams. The circular convolution operation involved in this requires a customised implementation of vector cross product ($O(D^2)$) that is used $n-1$ times for each n-gram constructed. This is computationally intensive, and largely unfeasible to run on the big unlabelled corpora used in this thesis; thus, only the BEAGLE context encoding is employed in the experiments.

HAL is an older and more widely applied model, for example by Lund and Burgess (1996); Burgess et al. (1998); Burgess and Conley (1998); Azzopardi et al. (2005); Song and Bruza (2001); Rohde et al. (2005); and therefore has been subject to more analysis. Rohde et al. (2005) give a comparison of HAL to other models, including the WordNet-based (Fellbaum et al., 1998) approaches (which are out of the scope of this thesis, because there is no equivalent resource for biological texts). They also built an extension to the HAL model that attempts to eliminate two perceived drawbacks of the original method: skewing of the context importance towards highly frequent basis and the noisiness added by low frequency terms. These concerns are also addressed by the information-theoretic methodology described in Song and Bruza (2001) for the task of query expansion in search engines; however, this approach does not transfer well to the matrix representation of the data. Rohde et al. (2005), on the other hand, filter low-frequency basis by removing the corresponding columns or by combining the information in them using SVD. The high-frequency terms are balanced using a smoothing algorithm that weights the basis corpus frequency with its importance to a particular target, an idea that is also explored by fully the probabilistic semantic model proposed by Dagan et al. (1999). The Rohde et al. (2005) model was shown to be more effective than HAL on traditional evaluation tasks and, likewise, for inferring statistical information in a smaller corpus of child speech (Riordan and Jones, 2007). It is likely that BEAGLE, like HAL, would be sensitive to the overall frequency of the basis. The peaks of the highly frequent basis words would be more prominent than the peaks of moderate words, while the low frequency terms would be masked by the lower peaks of the high frequency basis.

The basic versions of HAL and BEAGLE are used in this thesis in order to enrich kernel classification of PPIs. Both of these algorithms come with a space of options to be investigated, for example: the effects of training on different corpora with different features, window lengths or dimensions. These are explored in Chapter 6 and form a baseline for any future work that would explore other models or improvements in model smoothing. These models, with their best settings, are then combined to train a single

classifier, in Chapter 7.

Part II

Experiments

Chapter 5

Results of Supervised PPI Classification

In this chapter the Gaussian process (GPs) classifier is evaluated against two algorithms that are commonly used for text classification, the support vector machine (SVM) and the naïve Bayes (NB) classifiers. The aim of the experiments is to extensively compare the algorithms by evaluating them on several datasets, with different kernels and features. The products of this initial analysis are the optimal settings for each dataset, kernel, and algorithm. The best results from this chapter are used in the next two chapters as the baseline.

The following evaluative experiments involve a search across different datasets for the best features and algorithm settings. In order to conduct a most thorough comparison, each configuration is tested using the mean and standard error of evaluation measures (AUC and F-score) gathered from ten ten-fold cross-validation (10x10cv) experiments. Five corpora are tokenised in six different ways to produce feature sets F1, F2, ..., F6, which we then use in the classifiers. Of these five, two have gold-standard protein annotations, and three do not. From the diagram of the search space, shown in Figure 5.1, we can see that, after NER is applied, the post-processing of corpora effectively results in twelve different datasets. GPs and SVMs are tested with both cosine and Gaussian kernels. NB and the GP with cosine kernel only require one 10x10cv experiment, while the Gaussian kernel has one parameter (θ) which needs to be tuned. The SVM also has

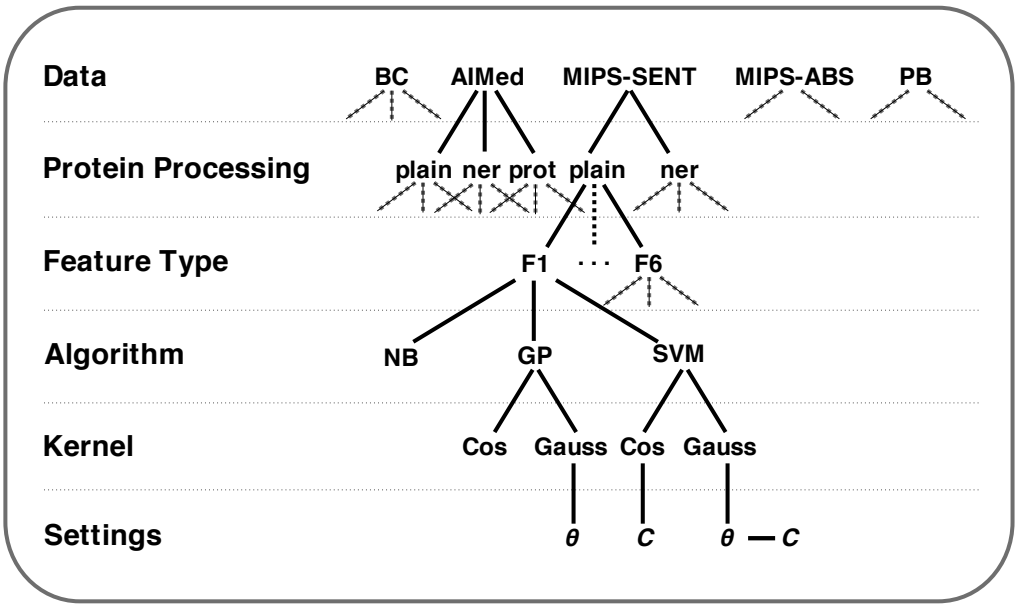


Figure 5.1: Graphical representation of the experimental search space, showing an example path leading to the leaves of the tree. Each leaf represents a 10x10cv experiment.

the margin parameter C that needs to be set. Therefore, by testing ten values for θ and ten more for C, there is a total of 26,352 10x10cv experiments. The datasets, protein annotation extraction, feature types, and results are described below.

5.1 Datasets and feature extraction

In this evaluation there are three sentence corpora, BC (BioCreative PPI), AIMed, and MIPS-Sent, as well as two abstract corpora PB (PreBIND) and MIPS-Abs, which were initially described in Section 2.2.2.

Table 5.1 gives an overview of the different corpora, including the ratio of documents that contain interactions (positive) in proportion to the size of the whole dataset. BC and PB contain labels for the positive examples, but AIMed contains a set of abstracts where the labelling indicates an interacting pair of proteins. So the sentences that con-

Corpus Name	Number of Documents	Percentage Positive	Average Doc Length (in features)	Protein Annotation	Number of Annotated Proteins per Document	Number of NER Proteins per Document
BC	999	17%	12.75	yes	3.68 (pos) 0.56 (neg)	1.77 (pos) 0.50 (neg)
AIMed	1980	31%	13.58	yes	3.33 (pos) 1.48 (neg)	2.70 (pos) 1.51 (neg)
MIPS Sent	4000	11%	13.22	no	—	1.32 (pos) 0.75 (neg)
MIPS Abs	4000	9%	108.28	no	—	22.9 (pos) 6.20 (neg)
PB	1081	63%	111.08	no	—	15.7 (pos) 8.01 (neg)

Table 5.1: Corpora statistics

tain at least one interacting pair are considered positive examples, and the ones which do not are considered negative. While BC, AIMed, and PB contain all the documents from their respective corpora, MIPS-Sent and MIPS-Abs are subsets of MIPS, which contains examples of both positive and negative abstracts. A random set of 4000 examples, for both sentences and abstracts, was selected from the whole MIPS corpus, preserving the original ratio of the data. The MIPS-Sent dataset has a slightly higher percentage of positive examples than MIPS-Abs, indicating that there is on average more than one positive sentence per abstract.

All of the corpora are annotated for named entities, the resulting text is considered a separate corpus in the experiments. Similarly the versions of the BC and AIMed corpora that contain the hand annotated protein names are considered separate inputs into the tokeniser. The tokeniser extracts all unique strings, based on a set of specified regular expressions. The output can also be processed by a stemmer. The unique token strings are indexed, and these index numbers are used to refer to cells in the document vectors. The feature extraction process is described in more detail in the rest of this section.

5.1.1 Protein named entities as features

Presence of proteins is one of the indicators of an interaction; however, protein names are highly variable, and thus may add noise to the feature set. By replacing the protein names with a string indicating the presence of a protein, the documents are turned into generalised patterns.

The performance of classifiers on the NER tagged data is compared against the plain data, which contains the original protein name strings. The BC and AIMed corpora allow

for comparison of the NER tagging versus the human protein annotation for the PPI sentence classification task. Hand annotated data can be used for training, but NER would still need to be employed if the model is used on new text. Consequently, to facilitate this comparison, all of the corpora were preprocessed with an automatic NE annotation tool, Lingpipe (Baldwin and Carpenter, 2008). This tagger was trained on the GENIA corpus (Kim et al., 2003) and recognises several types of named entities. Out of these, there are a few that are related to proteins including *protein_molecule, protein_family_or_group*, and *protein_complex*.

Evaluation against the AIMed dataset showed that using *protein_molecule* (*pm*) was closest to the hand-annotations, while Polajnar et al. (2009b) show that for some datasets other combinations of GENIA annotations may work better. Full testing of how different NER annotation schemes affect the classification of each of the datasets would add another dimension to the already large search space, and was thus omitted.

A preliminary evaluation was performed against the protein annotations in the AIMed corpus. The results show that the proteins were being located with high precision, although the annotation schemes were not well aligned. For example, the strict comparison, where the protein tags have to be perfectly aligned, demonstrates that the proteins were being located with high precision (P=0.7111), but lower recall (R=0.4764), leading to a fairly low F-score (F=0.5705); however, permitting partial matches increases both precision and recall by 0.12, thus raising the F-score (P=0.8359, R=0.5937, and F=0.6943). Partial matches are still a an accurate way of assessing the fitness of the tagger for this problem, because the classifier only considers the number of proteins that are in the document. Poor alignment between the NER and hand annotations affects the classifier only by varying the inclusion of the features that are surrounding the protein names or are contained within them.

Partial matches occur because protein names often span several words, can be nested, and there is no standard annotation protocol. Table 5.1 shows the average number of proteins found in the documents of each of the corpora by NER. It also shows the same

statistics for protein annotations in BC and AIMed. It is clear that the NER is much more in agreement with the AIMed corpus, and that perhaps for BC a looser interpretation of a protein name, for example using both *protein_molecule* and *protein_family_or_group*, might lead to a closer match. In fact, for BC and MIPS-Sent, NER detects less than two proteins per positive sentence on average, indicating that many interacting proteins are not being accurately annotated. For the abstract datasets, the protein counts show that there is a much larger number of proteins occurring per document, although there is still a large difference between the number of proteins in positive versus the negative abstracts.

A typical sentence after NE annotation is shown below:

```
We have identified a new <ENAMEX TYPE="protein_molecule">TNF - related ligand</ENAMEX> ,
designated <ENAMEX TYPE="protein_molecule">human  GITR ligand</ENAMEX> (
<ENAMEX TYPE="protein_molecule">hGITRL</ENAMEX> ) , and its
<ENAMEX TYPE="protein_family_or_group">human receptor</ENAMEX>
(  <ENAMEX TYPE="protein_family_or_group">hGITR</ENAMEX> ) , an ortholog of the recently
discovered  <ENAMEX TYPE="protein_family_or_group"> murine  glucocorticoid - induced TNFR -
related ( mGITR ) protein </ENAMEX> [ 4 ] .
```

The resulting annotations translate into features through substitution of a place-holder string `PTNGNE`, concatenated with the protein index, for the words comprising the NE. The index is a counter from 1 to the total number of proteins in the document, where each occurrence of a protein in a document is counted as unique unless it is enclosed in parentheses following another protein. In this case the simple algorithm assumes that both tagged entities refer to the same protein, but it does not keep track if the same protein occurs twice in different parts of the document. The final features extracted are:

```
identified  ptngne1 designated ptngne2 ptngne2 human receptor ortholog recently discovered
murine glucocorti induced tnfr related mgitr protein
```

The same sentence with the original hand annotations is:

```
We have identified a new TNF - related ligand , designated human <p1  pair=2 >
<prot>  <p1  pair=1 >  GITR </p1>  ligand </prot>  </p1>  ( <p1  pair=3 >  <p2
pair=1 >  <prot>  hGITRL </prot>  </p2>  </p1>  ) , and its human receptor ( <p2
  pair=2 >  <p2  pair=3 >  <prot>  hGITR </prot>  </p2>  </p2>  ) , an ortholog
of the recently discovered murine <prot>  glucocorticoid - induced TNFR - relate
d ( <prot>  mGITR </prot>  ) protein </prot>  [ 4 ] .
```

and results in the following features, disregarding protein name nesting:

```
identified tnf related ligand designated human ptngne1 ptngnel human receptor
ptngne2 ortholog recently discovered murine glucocorti induced tnfr related ptngne3
```

The preprocessed corpora are treated as new data sets and are tested with all the different feature extraction methods mentioned in Section 5.1.2.

As can be seen in Appendix A protein name features do not always lead to the highest performance. For BC and AImed sentence data using hand annotated proteins increases the AUC up to 8% and NER up to 6%. A generalisation cannot be made across all of the sentence data, because for MIPS-Sent the NER AUC is 3% lower than when no annotation is used. Likewise, the results are algorithm dependent, for example, on MIPS-Abs and MIPS-Sent, SVMs benefit from NER features, while GPs get slightly higher AUC on plain features.

5.1.2 Feature extraction

The way that words are extracted from the corpora controls how much information is preserved and this can in turn impact on classification performance. To test the amount of useful information contained on average in the biomedical words, several extraction techniques were considered. Shortening words has the effect of grouping some features together. Doing so automatically could either reduce or improve the performance of classification, because it applies the same discrimination criteria to all words.

Firstly, on the character level, two different tokenisation techniques are applied. These are referred to as *long* and *short*, and each in effect defines what constitutes a word. As discussed in Section 2.1.2, biological words can contain numbers and symbols that are not found in general domains such as news and emails. *Long* tokens are defined as strings that start with a letter, which can be followed by letters, numbers, dashes, or apostrophes. The other, much more conservative definition, only allows for letter characters. Thus, in the *short* tokens, if a word contains any other symbols after the letters, only the initial part

of the string is conserved. The effect is that, for example, words such as *IL-8* and *IL-10* get normalised to the token string *IL*.

Secondly, preserving the morphology of the word, including all the letters with their original capitalisation, results in a larger feature space, but might not add any more information at the document level. In one feature set the capitalisation is preserved, while in the rest all the characters are turned to lower case.

Finally, shortening the words also leads to grouping of some terms. One way of doing this is to penalise only the long words, *e.g.* Donaldson et al. (2003) keep only the first 10 characters of each word. Another way of truncating words is to apply stemming (Porter, 1997). Stemming shortens the words to their root so that, for example, all instances of the root *interact*, such as *interacts, interacting*, and *interaction*, would be normalised to the same string *interact*.

Six different combinations are used, each leading to further abstraction and shorter feature sets:

- **F1:** Long words with original capitalisation and full word length
- **F2:** Long words with original capitalisation and length truncated to 10
- **F3:** Long words with lowercase, truncated to 10 letters
- **F4:** Long words with lowercase and stemming
- **F5:** Short words with lowercase, truncated to 10 letters
- **F6:** Short words with lower case and stemming

Figure 5.2 provides an overview of results across the different datasets and algorithms for each of the feature types. In general, the feature types that offer higher abstraction, *i.e.* represent several of the original words, tend to have higher performance. The highest scores tend to be with stemmed long words; however, the lowest variance tends to be when numbers and symbols are ignored and words are truncated to 10 characters, and

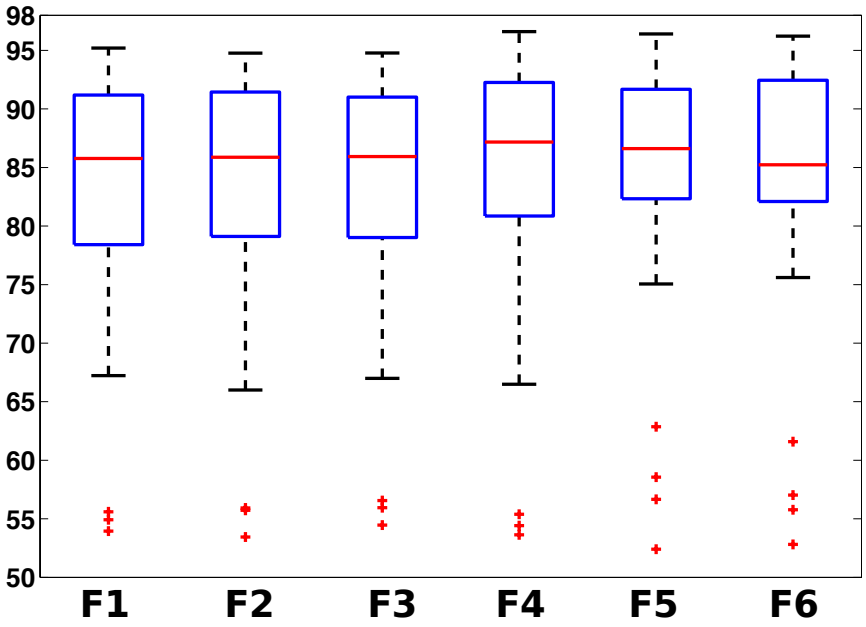


Figure 5.2: Comparison of performance for different feature types (F1-F6) across all of the different corpora and algorithms, listed on the x-axis. The y-axis represents the AUC values. The top of the boxes indicates the top 75% of the values and the bottom shows the lower 25% of the values. The horizontal line through the box shows the median value, the bars indicate the span of the values not considered to be outliers, while those are shown as separate crosses. This graph was generated by the MATLAB boxplot algorithm.

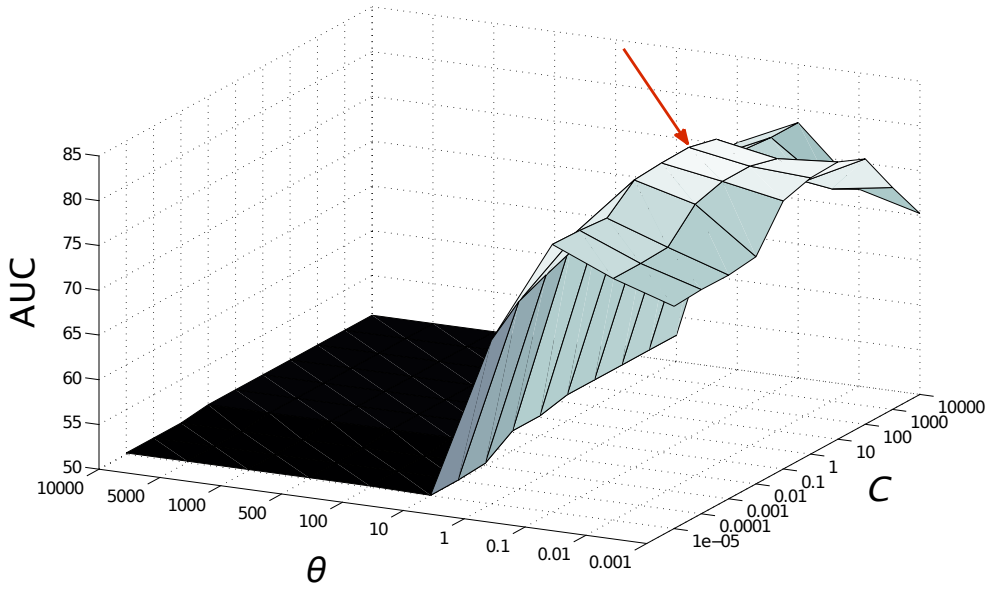


Figure 5.3: The AUC of different θ and C combinations for the SVM using the BC dataset with feature combination F1. The red arrow shows the point with the highest AUC of 82.42. The right choice of the kernel parameter is essential to classification, while the right choice of C allows the fine-tuning of classification accuracy.

thus this is the safest option to apply without testing for various feature types. The outlier results (shown by the + symbols) are some of the lower performing NB runs, these are more spaced out, and overall slightly higher for the short features F5 and F6. Table 5.2 contains the highest AUC scores for each of the datasets, and indicates which features and settings produced this highest result. This table almost exclusively contains feature types F4-F6, and a consistent pattern shows NER or protein features often coinciding with the long words with stemming.

5.2 Algorithm and kernel parameter selection

The choice of kernel parameters has a large impact on the GP performance; while for the SVM margin parameter C also needs to be tuned in conjunction with any chosen kernel parameters. The Gaussian and the cosine kernels (Section 3.2.2) are widely used in text classification literature. While the cosine kernel has no parameters, the Gaussian kernel

has a parameter θ, which needs to be tuned for each data set. For the SVMs the full range of C has to be tested for each value of θ. Thus, the SVM requires ten times as many tuning experiments for each data set as the GP. The evaluated ranges for the parameters are: $\theta \in$ (0.001, 0.01, 0.1, 1, 10, 100, 500, 1000, 5000, 10000) and $C \in$ (0.00001, 0.0001, 0.001, 0.01, 0.1, 1, 10, 100, 1000, 10000). These values are quite far apart and only provide an approximation of the magnitude of the best value.

Figure 5.3 illustrates SVM parameter tuning from the experimental results for one dataset-feature pair. The surface shows that θ controls the classification performance, while C provides a method for fine-tuning. Poor choice of the kernel parameters results in a poor distinction between the classes, *i.e.* all documents could be equally similar or constrained to a small range of distances. A poor choice of C, on the other hand, results in a bad placement of the hyperplane as shown in Figure 3.4. Improper grouping of the points in the space caused by the wrong choice of kernel or kernel parameter cannot be improved by a good choice of hyperplane. A good grouping of data can make data more separable, and thus even a badly tuned choice of hyperplane can provide some separation in the classes.

Figure 5.2 and Appendix A show that overall for unnormalised data both GPs and SVMs tend to gravitate towards lower values of θ, with GPs settling in on 0.01 for sentence and 0.001 for abstract datasets. SVMs also prefer 0.001 for the abstracts, but tend to vary between 0.1 and 0.001 for the sentence data.

The naïve Bayes classifier, which is used as the baseline Bayesian classification example, has no kernel, and thus no kernel parameters. As such, the NB cannot be used in the further experiments with semantic kernels. In addition, He and Ding (2007) show that varying the smoothing only gives a small improvement in classification accuracy; therefore, NB is not tuned for each of the datasets, and the choice of Laplace smoothing is considered sufficient.

5.2.1 Results

The experiments performed on the three classifiers cover many settings, which were discussed in detail in the above sections, and are summarised here. The following settings come from feature extraction:

- There are five datasets. In three of these (BC, AImed, MIPS-Sent) the PPI can be located at sentence level, while in the other two (MIPS-Abs, PB) they are only annotated at the abstract level. BC, AImed, and PB are higher quality datasets because they are annotated by human evaluators, while the MIPS dataset derivatives are annotated automatically.

- Each of these five datasets is further annotated for protein names using automatic NER. BC and AImed also contain protein name labels provided by annotators. Therefore, each of the datasets can be translated into plain, NER, and possibly protein features.

- The features types are also affected by the choice of tokenisation and whether the words are truncated by length or stemming.

A range of settings is also explored for the GP and SVM classifiers. These experiments seek to elucidate whether the Gaussian or the cosine kernel is more appropriate, and the effects of the Gaussian kernel parameter θ. For the SVM, the right settings for the margin parameter C are also required for highest classification accuracy. VBpMKL classification results are nearly identical to the GP, so in order to reduce the experimental search space it is only evaluated on the cosine kernel. Likewise, the choice of NB smoothing is not explored here.

Appendix A gives comprehensive tables of results, divided by kernel type, for each of the datasets with all the different feature types. The Gaussian kernel results are listed first, in Section A.1 and contain results for the NB classifier. The tables for the cosine kernel are listed in Section A.2 and also contain the results for VBpMKL.

A summary of the appendix is provided in Table 5.5, which shows the highest AUC for each of the datasets and the settings that were used to achieve this. An analysis on a per algorithm basis shows that in five out of the seven datasets the highest AUC is achieved by the SVM; however the GP scores are similar and much closer to the SVM than to the NB. Compared to the GPs and SVMs, the NB has particularly low AUC scores for the abstract data, showing that it has difficulty with the extended feature spaces of these datasets. Analysing the SVM and GP results by different kernel types shows that GPs, in general, perform better with the Gaussian kernel. In Appendix A, this trend can be observed in the tables containing the results for the BC, PB, and MIPS-Sent data, where the GP with Gaussian kernel consistently outperforms the SVM. The SVM has slightly higher AUC on the remaining datasets with the Gaussian kernel, and across most of the experiments with the cosine kernel. Figure 5.4 shows that there is no statistically significant difference between the GP and the SVM with the Gaussian kernel. Although the SVM has higher peak AUC, the middle 50% of the results are equivalent. The same figure shows that NB performs significantly worse than both of these algorithms. Similar boxplots for the cosine kernel, shown in Figure 5.5, demonstrate no difference in VBpMKL AUC across all the cosine results, as compared to the GP, but the SVM slightly outperforms both of the algorithms. These higher AUC values provide the SVM with slightly higher performance over all of the experiments (Figure 5.6).

Table 5.2 can also be examined by dataset. For the BC and AImed, the results table includes both the highest AUCs overall, which come from using the hand annotated protein features, as well as the highest scores obtained with the NER features. Substituting protein names by placeholder strings leads to an improvement in AUC for BC and AImed sentence datasets. These two corpora were carefully compiled and annotated and consequently they produce good quality models. Using hand-annotated protein names leads to a larger improvement on BC data, while for AImed there is very little difference between using automatic and human annotations. There is a higher correlation between the number of proteins found by these two methods for the AImed dataset, as can be seen

GP				
Data	**Features**	**K**	**Settings**	**AUC**
BC	NER + F4	G	θ=0.01	87.62 $\pm$ 0.42
BC	PROT + F4	G	θ=0.01	92.27 $\pm$ 0.24
Almed	NER + F4	G	θ=0.01	89.25 $\pm$ 0.22
Almed	PROT + F4	G	θ=0.01	90.24 $\pm$ 0.20
MIPS-Sent	F5	G	θ=0.01	86.90 $\pm$ 0.27
MIPS-Abs	F6	C	—	95.31 $\pm$ 0.15
PB	F6	G	θ=0.001	93.34 $\pm$ 0.25

SVM				
Data	**Features**	**K**	**Settings**	**AUC**
BC	NER + F4	C	C=0.1	87.44 $\pm$ 0.5
BC	PROT + F4	C	C=1	93.13 $\pm$ 0.28
Almed	NER + F5	C	C=1	90.83 $\pm$ 0.30
Almed	PROT + F6	C	C=1	92.76 $\pm$ 0.17
MIPS-Sent	F5	G	θ=0.1, C=1	86.53 $\pm$ 0.29
MIPS-Abs	NER + F4	C	C=10	97.29 $\pm$ 0.16
PB	NER + F6	C	C=1	94.03 $\pm$ 0.23

NB		
Data	**Features**	**AUC**
BC	NER + F4	83.97 $\pm$ 0.48
BC	PROT + F4	86.80 $\pm$ 0.47
Almed	NER + F4	87.11 $\pm$ 0.22
Almed	PROT + F4	83.60 $\pm$ 0.29
MIPS-Sent	F6	83.01 $\pm$ 0.35
MIPS-Abs	F5	56.66 $\pm$ 0.58
PB	NER + F1	67.23 $\pm$ 0.54

Table 5.2: Results table where the feature settings are indicated by the F-measure, kernel choice is in column K where G represents the Gaussian and C the cosine kernel. The settings column shows the θ and C that lead to the best performance.

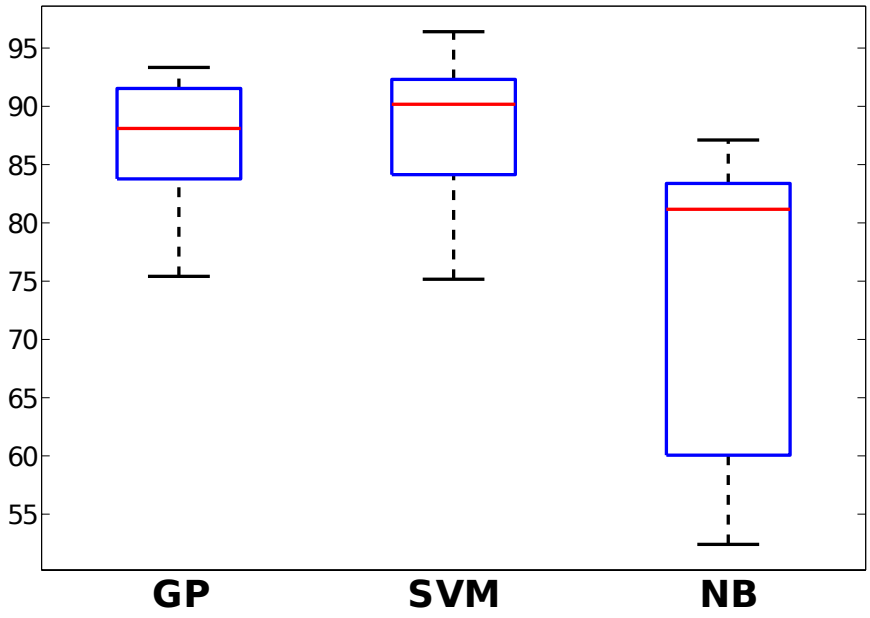


Figure 5.4: Comparison of GP and SVM with the Gaussian kernel and NB, across the different corpora and feature types. The y-axis represents the AUC. The t-test p-value of 0.3339 shows that the difference between the SVM and the GP is statistically not significant, while the NB is significantly worse than the GP with p-value of $6.1062e^{-15}$.

in Table 5.1. The lower classification performance of the NER features matches with the observation that, for the MIPS and BC corpora, the average number of proteins found by NER in positive sentences is less than 2. The number of proteins an NER finds per positive sentence may be a good indicator of how well it will perform as a feature extraction method for classification.

BC, AImed, and PB are high quality hand-annotated datasets. The MIPS data consists of abstracts that were confirmed as relevant or irrelevant during a database curation process; however, the sentence annotation was done by automatically choosing the sentences that contain the interactants. Both MIPS-Abs and MIPS-Sent are samples from a large dataset, containing 4,000 abstracts and 4,000 sentences respectively, and thus are larger than the three meticulously collected corpora. MIPS-Abs leads to higher AUC with the kernel classifiers than the smaller abstract dataset, PB; however, it also has a much larger feature space, which reduces the naïve Bayes effectiveness. The MIPS-Sent data, on the other hand, leads to lower classification performance than the BC and AImed corpora,

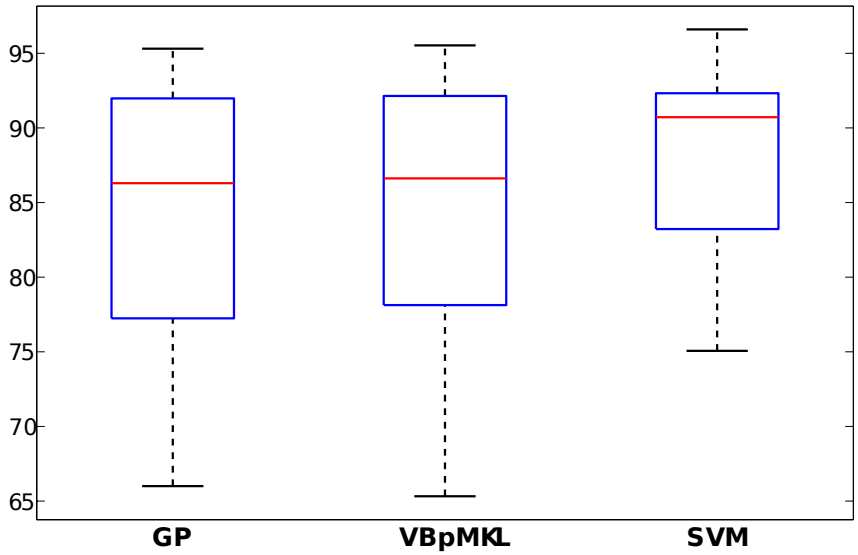


Figure 5.5: Comparison of GP, VBpMKL, and SVM on the cosine kernel across different corpora and feature types. The y-axis represents the AUC. The t-test indicates that the difference between GP and VBpMKL is not statistically significant ($p = 0.9115$); however the SVM are marginally, but significantly better than GPs (p =0.0011).

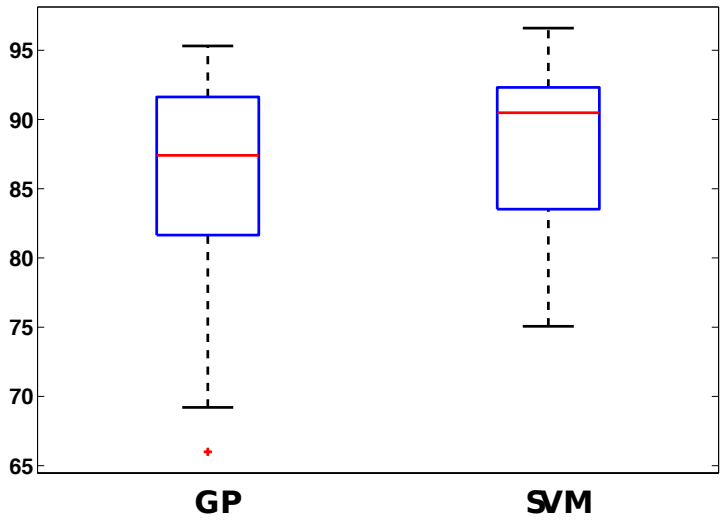


Figure 5.6: Comparison of GP and SVM across both cosine and Gaussian kernels. The y-axis represents the AUC. Overall the SVMs better performance of the SVMs with the cosine kernel ensures that the difference between the algorithm is statistically significant ($p = 0.0015$).

although it is much larger. This implies that the quality of the data is more important for model training than the quantity.

5.3 Related Experiments

In this section, a broad set of experiments is presented to expand on preliminary observations published in Polajnar et al. (2009b) and Polajnar et al. (2009a). In the initial experiments the algorithms were compared on features without numbers and punctuation, truncated to ten letters (F5); and the algorithm parameters were tuned to achieve a high F-score, whereas here they are tuned for high AUC. The initial findings showed that, depending on the dataset and feature type, the SVMs and GPs had close if not equivalent performance.

Polajnar et al. (2009b) also found that it is possible to train on one of the datasets and apply the model to the others while maintaining relatively high F-score. This type of experiment simulates the use of a model on new samples collected from a different set of queries on MEDLINE. Table 5.3 shows the results from this initial cross-corpus study. Using PreBIND for training of the GP classifier, and AImed for testing, produces a high recall, but low precision. The same combination produces a more even response in the SVM precision-recall balance. The area under the ROC curve (AUC), however, is the same between the two algorithms. Using NER features increases the AUC for both algorithms, a result that is also observable in the F-scores and accuracies.

On the other hand, reversing the training and testing corpora also causes the precision-recall relationship to be inverted; and while the NER features still increases the SVM F-score, the AUC for both classifiers decreases (from 0.75 to 0.70 for the GP and from 0.80 to 0.77 for the SVM). Considering an alternative NER scheme (NER2), which Polajnar et al. (2009b) found, produced a better cross-validation results for the PB dataset results in more effective training (as shown in Figure 5.7). NER2 is a scheme that considers the Lingpipe (Baldwin and Carpenter, 2008) NER annotation of *protein_or_family_group* also

Corpus		Features	GP			
Train	Test		F	AUC	P	R
PB	AImed	F5	0.54	0.72	0.38	0.94
AImed	PB	F5	0.21	0.75	0.98	0.12
PB	AImed	F5 + NER	0.70	0.79	0.57	0.95
AImed	PB	F5 + NER	0.15	0.70	0.97	0.08
AImed	PB	F5 + NER2	0.45	0.78	0.94	0.29
Corpus		**Features**	**SVM**			
Train	Test		F	AUC	P	R
PB	AImed	F5	0.57	0.72	0.42	0.86
AImed	PB	F5	0.57	0.80	0.93	0.41
PB	AImed	F5 + NER	0.69	0.82	0.57	0.88
AImed	PB	F5 + NER	0.62	0.77	0.89	0.48
AImed	PB	F5 + NER2	0.74	0.79	0.83	0.67

Table 5.3: Cross-corpora experiment results for GPs and SVMs. Each row shows whether the classifiers were trained or tested on the PreBIND (PB) or the AImed corpus and what features were used. The results are presented as F-score (F), AUC, precision (P), and recall (R). The results were obtained using the cosine kernel, which, as has been demonstrated in this chapter, is more effective when paired with the SVM than with the GP.

as an indication that the string may be a protein. It may be intuitive that this annotation scheme causes a larger number of proteins to be found in AImed sentences, and thus brings the total number of proteins per document closer to what is found in the longer PB documents (Table 5.1).

In general, cross-corpus experiments show a 10-15% drop in the AUC compared to the cross-validation results on the same datasets as shown in Appendix A. Therefore, as models trained on abstracts in general have higher AUC, training on abstract data leads to a more predictive model for classification of sentences, than vice versa. The differences in the distributions of the positive examples in the training data can lead to a skewed precision-recall balance, indicating that using the probabilistic output of the GP to rank the samples is more effective than assigning a class based on a threshold learned from training data.

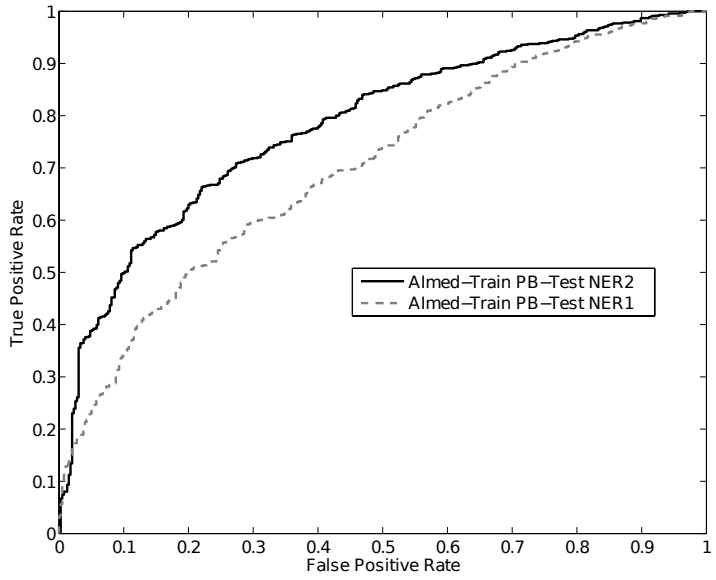


Figure 5.7: ROC curves demonstrating the effect that changing the named entity annotation scheme has on the cross-corpus testing AUC.

5.4 Discussion

The more comprehensive set of experiments detailed in this chapter confirm the results from Polajnar et al. (2009b) that show that GPs and SVMs are, for the most part, equivalent when classifying PPI sentences and abstracts. Across all of the experiments, the NB AUC scores are visibly and significantly worse than the GP and the SVM ones, while the SVM is slightly, although significantly higher than the GP. This difference in performance mainly stems from the experiments with the cosine kernel, where the SVM has higher performance; however the accurate tuning of the margin parameter C is the key to finding the best SVM performance. For example, Figure 5.8 shows how the variation in the margin parameter can drastically effect the classification AUC. Thus in most of the experiments, due to GP preference for the Gaussian kernel, and SVM preference for the cosine kernel, the two algorithms require equal amounts of tuning.

A high AUC shows that most of the positive documents have been rated higher than the negative ones. In addition, due to the skewed nature of the problem, *i.e.* there are many more sentences that do not contain evidence of PPI than the ones that do, evaluating

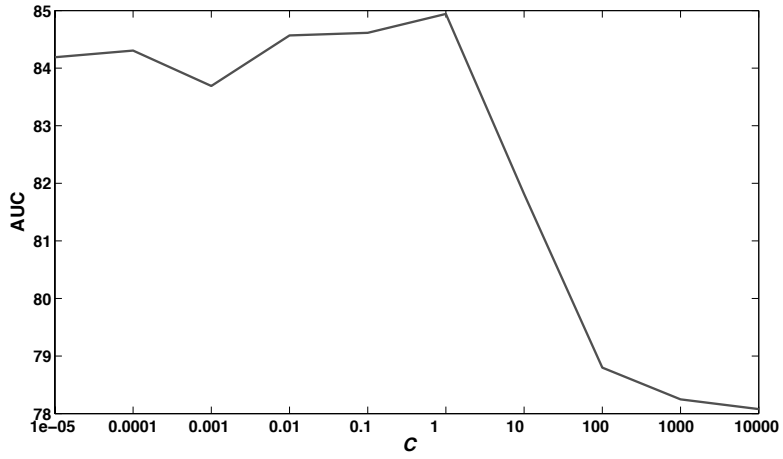


Figure 5.8: Tuning of the SVM margin parameter C for the cosine kernel (BC NER data with feature type F5).

the sentence ranking is more effective than class prediction. As was shown theoretically in Chapter 3, probabilistic models, such as GP, VBpMKL, and NB, provide a document ranking based on probability of class membership; while, the SVM ranking reflects the distance from the dividing hyperplane.

This chapter demonstrates that, overall, with the right choice of kernel, the GPs and SVMs have similar performance. That is, with the mean AUC of 87.34 for the GP (Gaussian kernel) and 88.00 for the SVM (cosine kernel) the difference between two algorithms is not statistically significant (p=0.53). In the following two chapters, the experiments exploring the semantic kernels will be performed using the GP and the VBpMKL algorithms. VBpMKL was shown to have performance nearly equivalent to the GP, and thus can be used instead of GPs for the kernel combination experiments in Chapter 7. Neither of these algorithms has a parameter which requires tuning alongside the kernel parameters, and therefore each will require less experimentation than the SVM, which has the margin parameter C. The best GP results from this chapter, as listed in Table 5.2, will be considered the baseline for these further experiments.

Chapter 6

Semi-supervised Learning through Semantic Kernels

In the previous chapter, the highest performance on each of the datasets was achieved by tuning each of the classification algorithms. Further improvements in the quality of the predictive models with these algorithms can only be gained by adding further training data. The plots for the NB and GP classifiers in Figure 6.1 show how the classification performance improves as more training data is added. Chapter 2 described the expense of obtaining high-quality PPI data annotation, prompting research into ways to gather information without relying on further, costly human assessments.

While quality labelled data is difficult to obtain in large quantities, unlabelled data is plentiful and freely available in the form of MEDLINE abstracts and full-text open access publications. Semi-supervised learning (SSL) (Chapelle et al., 2006; Abney, 2007) is a way to leverage the models trained on labelled data with large amounts of unlabelled data. This chapter describes a novel approach to semi-supervised learning, where information collected from relevant large datasets, in an unsupervised manner, is incorporated directly into the training kernel. The unlabelled corpus is transformed into a matrix of term similarities, which is then projected onto the document vectors causing a rescaling of the labelled training data.

The chapter first describes the traditional approach to SSL, to contrast with the novel semantic kernel method. This is described in Section 6.2, which begins by detailing two ways of integrating word co-occurrence statistics into the kernel, each producing slightly different scaling effects on the training data. The semantic information is gathered using HAL and BEAGLE, two methods described in Chapter 4. Their effects on the words in the training data are reported in Section 6.2.1.4. Section 6.3 describes experiments testing the semantic kernels in the classification setting, as well as the results of these tests. The classification is performed using the GP and VBpMKL classifiers, whose use is justified in the background Chapter 3 and the experimental Chapter 5. Further tests are then performed using LDA (Chapter 3) to discover topics within the best performing word-word co-occurrence matrix. This chapter concludes with the discussion of the methods and results.

6.1 Semi-supervised learning

Semi-supervised learning is typically performed by integrating the unlabelled data into the the labelled data for training. This approach, however, implies that the labelled and unlabelled data are drawn from the same distribution (Chapelle et al., 2006, Chap. 1). Although a dataset such as AImed comes from MEDLINE, the distribution of positive and negative examples is altered through the construction of the corpus. These corpora are constructed through careful selection of documents using various search queries and criteria, and therefore constitute specialised subsets. This can be seen from the corpus vital statistics shown in Table 5.1.

Figure 6.1 shows results from an experiment testing out Gaussian process (GP) classifier and naïve Bayes classifier SSL on the AImed data set. The entire data set is divided into ten portions for ten-fold cross-validation. The nine parts, which are used for training, are halved and one half has the labels removed. From the other half, we use n labelled documents, where $n \in \{[1-10], 20, 30, ..., 240, 250\}$. Figure 6.1 shows that initially

the model generated from one or two labelled points is not strong enough to support unlabelled data. As more labelled points are added, the unlabelled data can be better associated with similar labelled samples, and the GP SSL model improves over the simple GP which is only being trained on the labelled data. As the ratio of labelled to unlabelled points evens out the benefits of SSL also reduce, implying that a high proportion of unlabelled data is required[1]. Similar pattern can be seen on other datasets for both GPs and NB (Polajnar et al., 2009b).

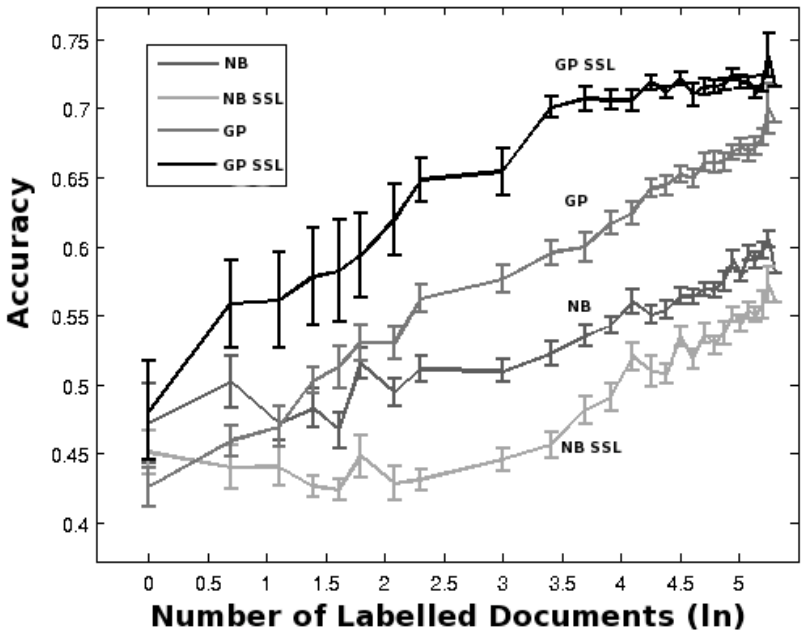


Figure 6.1: GP and NB semi supervised learning on AImed data. As the ratio of labelled documents increases the learning curve levels off. The number of labelled documents $n \in \{[1-10], 20, 30, ..., 240, 250\}$ is shown in log scale on the x-axis.

For example, for the PB corpus, the NB approach shows a similar improvement curve; however, for the AImed dataset the NB SSL fails and in fact, the introduction of unlabelled data produces a negative curve (Figure 6.1). This is due to the negative correlation between the predictive likelihood and classification accuracy, shown in Figure 6.2, and is a recognised problem with this algorithm (Nigam et al., 2006).

[1]This type of data may arise naturally in a PPI database construction scenario, provided all of the curation decisions are logged.

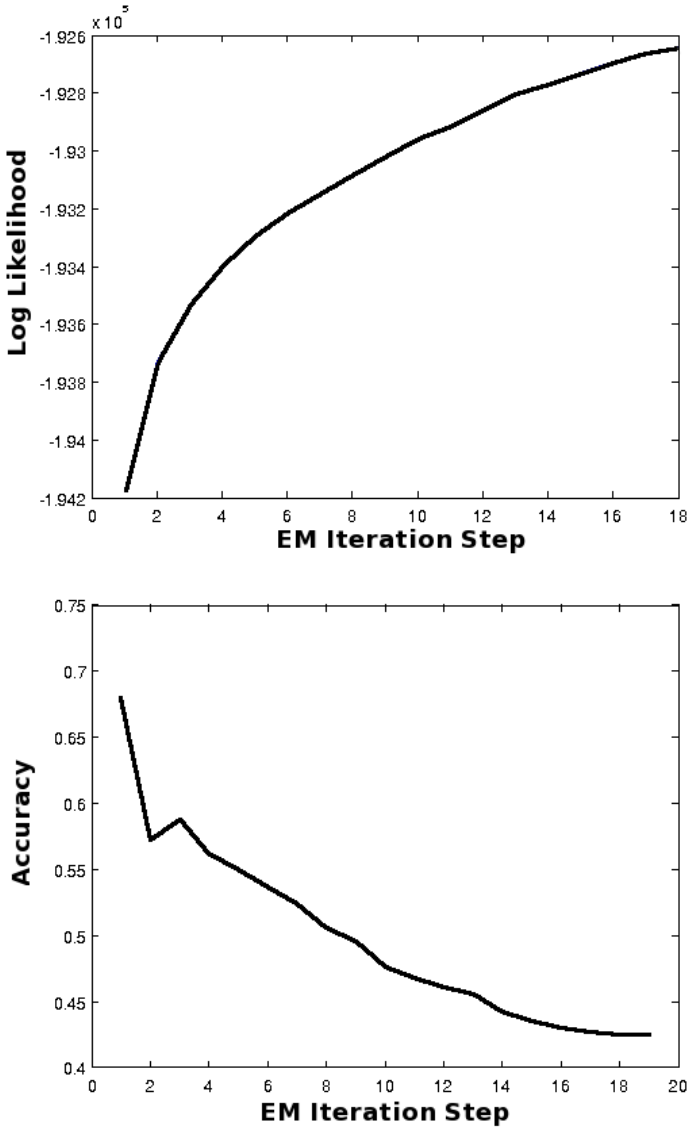


Figure 6.2: Negative correlation between log likelihood and accuracy for one CV fold of the NB SSL algorithm on the AImed dataset.

Therefore, while it is possible to see improvements in classification using SSL, it is not possible to do that without unlabelled data from the same distribution. In order to improve on the results from the previous chapter, a new method is required that forgoes that restriction.

6.2 SSL with semantic kernels

The novel way of combining labelled and unlabelled data, which is proposed in this thesis, integrates semantic information from unsupervised word co-occurrence models trained on a larger corpus not annotated for protein interactions, such as GENIA or the OAA.

The following procedure is used to enrich the kernels with semantic information:

- Initially, we have the labelled training data used in Chapter 5 to compare the different algorithms. This data is represented as a matrix $\mathbf{X}$ with M rows containing vectors representing the documents, whether they are sentences or abstracts. Each vector, of length N, contains the number of times each of the N features appears in the specific document; thus, the sparseness of the vectors is dictated by the number of unique features contained within the documents. Sentences, in particular lead to a quite sparse $M \times N$ matrix $\mathbf{X}$, while the abstracts are longer documents, and thus generally lead to a slightly more dense training data.

- The word ordering in the training data is lost due to using a bag-of-words representation. Consequently, the semantic meanings implied by the word's nearest neighbours are also lost. This information can be encoded by observing the usage of the features, which appear in the training data, across a larger corpus.

- A semantic model, such as HAL or BEAGLE, is used to collect word co-occurrence information from a large corpus of related biomedical content. The matrix returned by the models, $\mathbf{H}$ or $\mathbf{B}$ respectively, encodes the number of times a word occurs with other words within a specified context. The details of the construction of these

matrices are given in Chapter 4.

- Two ways in which the co-occurrence information is integrated into kernel classification in this thesis are:

  T1: Type 1 semantic kernel: A matrix of co-occurrence frequencies can be used directly to scale the training data matrix $\mathbf{X}$. Then any kernel transformations can be directly applied to this rescaled data.

  T2: Type 2 semantic kernel: In the co-occurrence matrices the semantic similarity between words can be calculated by applying a geometric distance metric. The cosine and Gaussian kernels are valid metrics and applying either of them to $\mathbf{H}$ or $\mathbf{B}$ results in a square matrix of word-word similarity scores, $\mathbf{S}$. In this second approach, this similarity matrix is used to scale the training data, $\mathbf{X}$.

The rest of this section describes the semantic kernel construction process in greater detail.

6.2.1 Semantic kernel construction

The semantic kernel is a constructed from two components, the $M \times N$ training data ($\mathbf{X}$) and the associated labels are used for classifier training and testing as before, but now the contributions of individual words in each document are rescaled by the semantic information from HAL and BEAGLE. Without application of any kernel transformations, the resulting combined space would translate into $\mathbf{XHH^TX^T}$ for HAL or $\mathbf{XBB^TX^T}$ for BEAGLE. This basic premise can be used to construct viable kernels in at least two different ways. The first way involves the transformation of the co-occurrence matrices into $N \times N$ word-distance matrices, while the second way uses the $N \times N$ HAL and $N \times D$ BEAGLE matrices to scale the training data before it is kernelised.

6.2.1.1 Semantic information collection

The HAL and BEAGLE matrices are constructed as described in Chapter 4. The HAL matrix contains the mutual co-occurrence frequencies between the features of the training dataset; alternatively, the BEAGLE matrix is constructed by considering any unique basis words that are not stopwords. Due to the random mapping scaling employed in BEAGLE, thousands of basis words can be represented in a matrix with a reduced number of dimensions D. Employing two different approaches in gathering of frequency counts also gives two different views of the data. In addition, through experimentation it was found that smoothing the data matrix as well as HAL matrices with a small number $\epsilon = 0.01$ leads to better classification performance. For BEAGLE, this smoothing is unnecessary because the matrices are not sparse.

6.2.1.2 Type 1 semantic kernel (T1)

The T1 kernel is created in two steps. The first step consists of transforming a co-occurrence matrix into a word-word similarity matrix. The semantic distance between words is calculated by applying a Cartesian distance metric to every pair of target vectors. Cosine distance is one of the most commonly used measures used to evaluate word distance in a vector space (Padó and Lapata, 2007); so instead of using the pairwise comparison of vectors, we can apply the cosine kernel (Equation 3.1) to the HAL or BEAGLE matrix and find the similarities between all of the words at once. This can be done with any kernel function, and thus the experiments in this chapter also evaluate the use of the Gaussian kernel (Equation 3.2) for this purpose.

As the procedure for construction of the kernel can be applied to $\mathbf{H}$ or $\mathbf{B}$ interchangeably, the examples here will use the HAL matrix for illustration purposes. Hence, by applying a kernel transformation to $\mathbf{H}$ we get a valid kernel $\mathbf{S} = \kappa(\mathbf{H}, \mathbf{H})$. Using the rules of kernel construction (Section 3.2.2), a new kernel can be created by inserting a

kernel, of correct proportions, into the inner product calculation:

$$\mathbf{K}_{T1} = \mathbf{X}\kappa(\mathbf{H}, \mathbf{H})\mathbf{X}^T = \mathbf{X}\mathbf{S}\mathbf{X}^T \tag{6.1}$$

Figure 6.3 shows a cosine kernel of $\mathbf{X}$ and then the same data subsequently scaled by $\mathbf{S}$ created by passing a HAL matrix through a cosine kernel. The kernels are normalised so that all the diagonal elements are 1. The result of the scaling process is that the most of the sentence similarities, which predominantly range 0 to 0.30, have been increased to range between 0.5 and 1. Thus, not only were the similarities amplified, but the range of similarity was also widened. The transformation is less visible on the Gaussian kernels, which produce higher sentence similarity values without the scaling.

6.2.1.3 Type 2 semantic kernel (T2)

In most cases the T1 semantic kernel construction method gives the best performance; however, for BEAGLE cosine transformations scaling the data before kernelisation is more effective. The following equation describes this second method:

$$\mathbf{K}_{T2} = \kappa(\mathbf{X}\mathbf{H}, \mathbf{X}\mathbf{H}) \tag{6.2}$$

Figure 6.4 shows that the same increase in sentence similarity values is produced by T2 as with T1. In addition it shows a clearer separation of the first 173 positive documents.

6.2.1.4 Effects of the semantic kernels

The effect that the similarity matrices have in rescaling the original training data are best observed through an example. We have the sentence i from the BC dataset:

```
PTNGNE1 by itself did not activate PTNGNE2 , PTNGNE3 and PTNGNE4 ( PTNGNE4 ) ,
whereas PTNGNE5 directly enhanced the PKC-dependent activation of PTNGNE6 induced
by other agonists including PTNGNE7 and phorbol esters , without affecting the
PTNGNE8 activation by those agonists
```

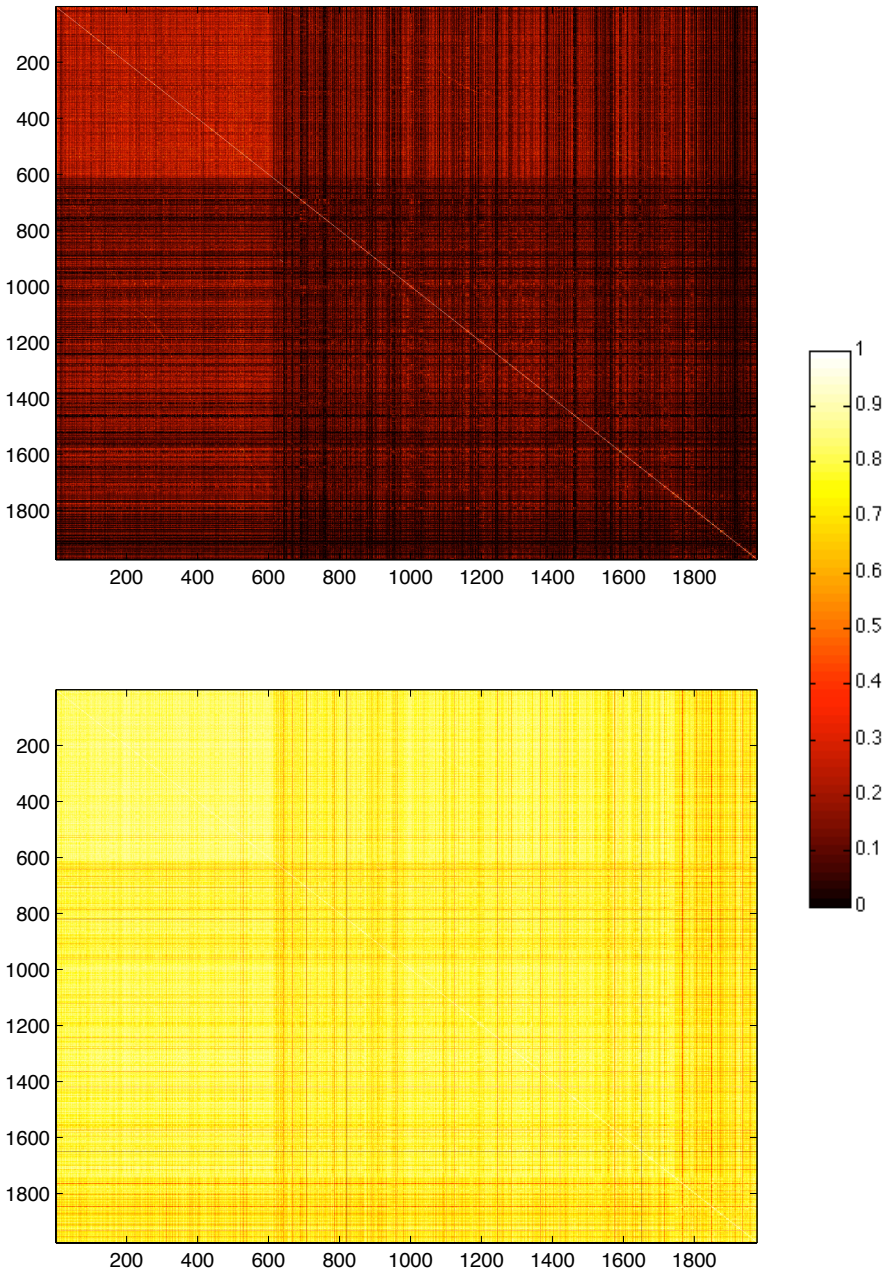


Figure 6.3: The cosine kernel (top) and the T1 HAL cosine kernel (bottom) of the AImed data. Both the x and y axes represent the documents from the collection. The first 614 documents are positive, the rest are negative. Introduction of the semantic information increases the similarity between documents, as evidenced by the colours in the kernels.

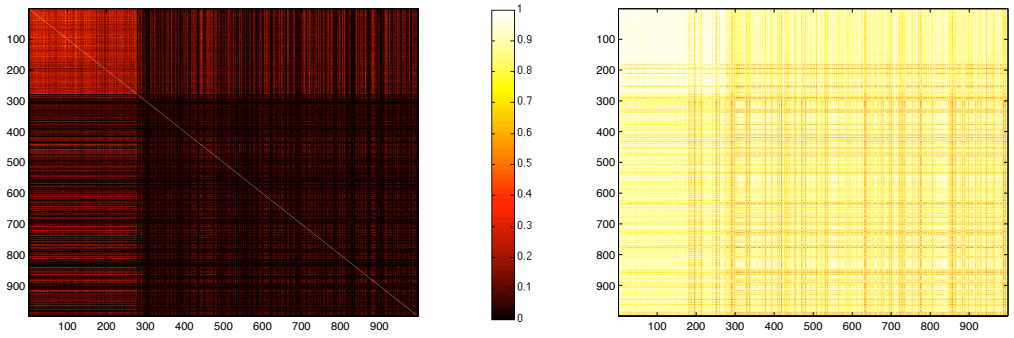
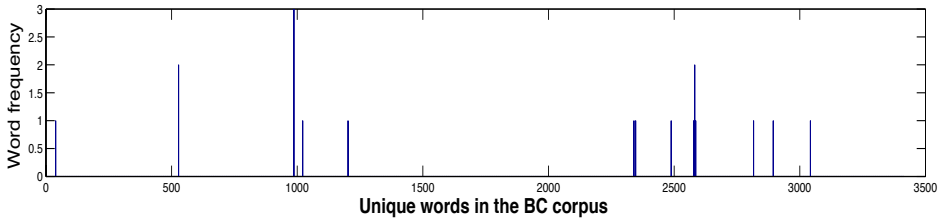


Figure 6.4: The cosine kernel (left) and the T2 BEAGLE cosine kernel (right) of the BC data. Both the x and y axes represent the documents from the collection. The first 173 documents are positive, the rest are negative.

which yields the following stemmed features (F5), shown in the order which they appear in, in the internal dictionary:

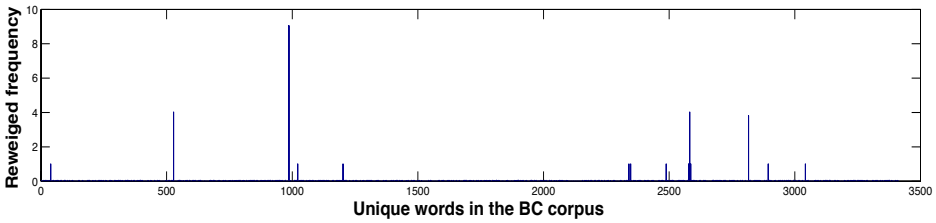
```
affect agonist activ enhanc directli ester wherea includ ptngne1 ptngne3
ptngne2 ptngne5 ptngne4 ptngne7 ptngne6 ptngne8 pkc-depend phorbol induc
```

This can be represented as a vector $\mathbf{x}_i$, so that each of the words that occurs in the sentence has a non-zero integer value indicating the number of times that word occurs in the sentence. That vector can be visualised as a bar graph, where the x-axis represents each of the unique words occurring in the corpus, while the y-axis represents the frequency of those words within the sentence i. The above words, can be seen as the spikes in such a graph, with the first spike being `affect` and the last spike being `induc`:



There is also a cluster of spikes around index 2600. These are the `ptngne` features are grouped together. These graphs are an easy way to visualise how features are reweighed through multiplication with the semantic kernel, causing a weighted inner product which boosts some features, whilst reducing others. The sentence similarity $\mathbf{x}_i\mathbf{S}\mathbf{x}_j$ will yield a single number; however, the rescaling is best visualised through the dot product $\mathbf{x}_i\mathbf{S}\cdot$

$\mathbf{x}_j$. The sum of this product is equivalent to the similarity value before normalisation, $\mathbf{x}_i\mathbf{S}\mathbf{x}_j = \sum \mathbf{x}_i\mathbf{S} \cdot \mathbf{x}_j$. The following is a visualisation of the sentence self-similarity $\mathbf{x}_i\mathbf{S} \cdot \mathbf{x}_i$:

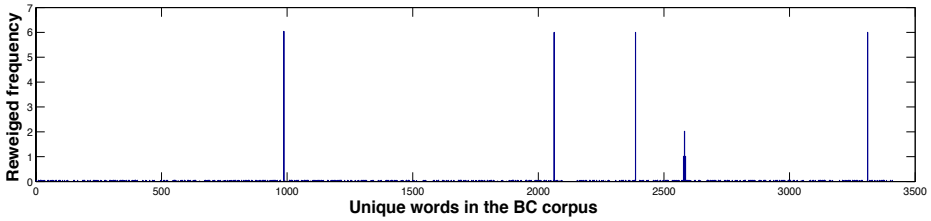


While the following two graphs show how the sentence similarities are calculated with, first, a highly similar positive sentence, and then a negative sentence. The positive sentence contains the following features:

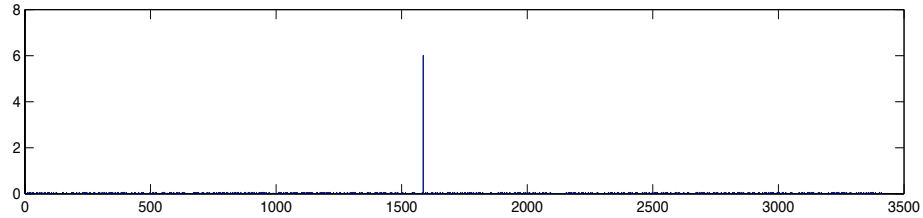
```
full-length report result human fusion activ domain coexpress rna-binding-defici k296r
ptngne1 ptngne3 ptngne2 ptngne5 ptngne4 ptngne7 ptngne6 ptngne8 mous doubl wild-typ
mutant dna-bind catalytic-defici
```

out of which the following have a weighting of more than 0.5 in the graph below:

```
activ rna-binding-defici k296r ptngne1 ptngne3 ptngne2 ptngne5 ptngne4 ptngne7
ptngne6 ptngne8 catalytic-defici
```



Similarly, the negative sentence features `diagnosi, haemochromatosi,` and `serum-ferritin` get re-weighted leaving only `serum-ferritin`:



The semantic kernel, in effect, applies the external knowledge of feature usage to increase or lower the importance of certain features. The more frequently a word occurs with the basis words, the higher the values in both its BEAGLE and HAL context vectors. Thus as the key scaling factor is the sum of the target vector, this example is valid for both HAL and BEAGLE, but also for both T1 and T2 kernel constructions.

6.2.2 Word similarity in biomedical texts

Similarity models are difficult to assess. For example, Figure 6.5 shows two versions of $\mathbf{S}$, both of which lead to an improvement in classification; however, the information which they provide is clearly different. In general, word co-occurrence models are evaluated through tasks, such as synonym pair recognition on a TOEFL exam, or more often by how well they appear to rank well-known words. For biomedical texts, there are no synonym lists, or a WordNet (Fellbaum et al., 1998) equivalent. Nevertheless, here are a few examinations into the effects of HAL and BEAGLE on biomedical texts through visualisations and qualitative evaluation of easily interpretable words.

The primary way in which the kernel $\mathbf{S}$ influences the data is through the cross product $\mathbf{XS}$. Here, the word weights are changed through multiplication between the word frequency in the sentence and the similarity vector of each of the words in the lexicon. The sentence features that are highly similar to many other terms in $\mathbf{S}$ will be boosted. A term is highly similar to others if the sum of the corresponding row in $\mathbf{S}$ is large. We will refer to this sum as the *similarity quotient* (SQ) of a term t, that is, $\mathrm{SQ}(t) = \sum \mathbf{s}_t$.

This measure is a way to provide an overview of the weightings assigned by a kernel. For example, Figure 6.5 shows that the cosine matrix is brighter than the Gaussian matrix. This phenomenon holds across the different experiments performed in this chapter for both HAL and BEAGLE. The cosine kernel has more low-level similarities across the whole lexicon, while the Gaussian kernel has sharper decline, some words are quite similar, or not similar at all. As a result, many words in a cosine kernel will have a quite high SQ, and in general these words seem to be the more commonly occurring ones. The features with high SQ in a Gaussian kernel tend to be the more rare ones, such as protein names. This can be seen in Table 6.1 (c) where high SQ words for both kernel types are listed.

Consequently, the Gaussian kernel weightings can be used to find relations between relatively rare terms such as protein names, as shown in Table 6.1 (b). In several HAL-based experiments, the performance of classification increases with the window length

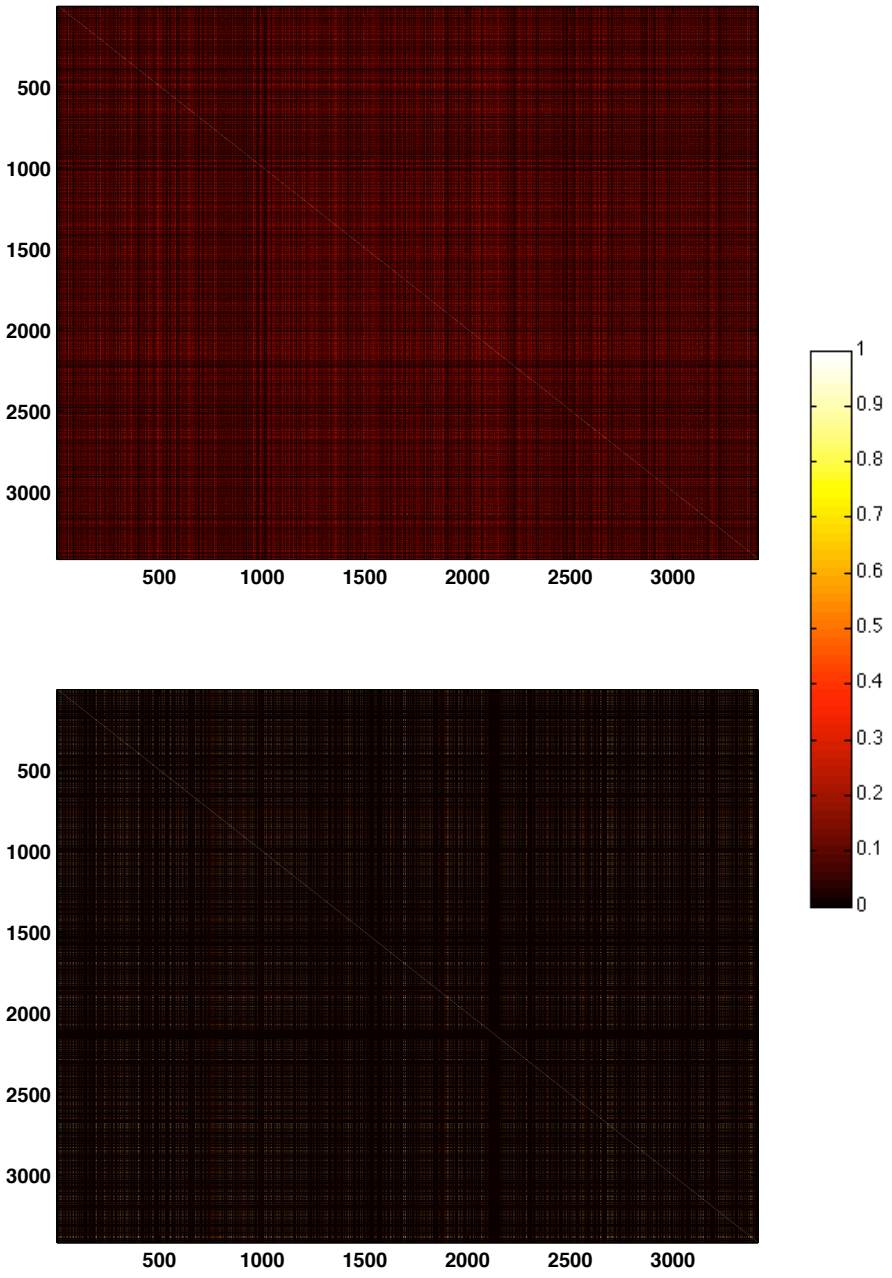


Figure 6.5: The cosine kernel of the $\mathbf{H}_{L=1}$ (top) and the Gaussian kernel of the $\mathbf{H}_{L=5}$ (bottom) of the BC words co-occurrence data as collected from the OAA. Both the x and y axes represent the unique words in the collection. The cosine kernel gives very little discrimination between the similarity values assigned to the words. Most of the non-zero values are in the red and orange range of the scale. The Gaussian kernel gives less similarity overall, but a greater distinction between the highly similar and somewhat similar items, using the full spectrum of values.

(a)

ptngne1	ptngne9
ptngne1	ptngne9
ptngne3	ptngne7
ft3	modul
respect	receptor
arginyl	mediat
gmt	enhanc
downregul	nuclear
antigen	activ
enhanc	function
protein	endogen

(b)

TNFalpha
TNFapha
junB
erythroleukemic
ld3
Hox11
CD8alpha
LIGHT
gal
cd44
perforin

(c)

Cosine	Gaussian
specif	cotransduc
depend	xcid
direct	copatch
variou	semist
determin	ec12
potenti	hlh462
modifi	viscosimetr
form	y429
possibl	anticalexin
indic	deltah

Table 6.1: Similar words from different feature types: (a) Short, stemmed words from the BC corpus with similarities from the $\mathbf{H}_{L=1}$ cosine kernel matrix. (b) Long features from the AImed corpus with similarities from the $\mathbf{H}_{L=4}$ Gaussian kernel matrix. (c) The short, stemmed words from the AImed corpus with highest the similarity quotient from the $\mathbf{H}_{L=9}$ matrix with both Gaussian and cosine kernels.

L. This indicates that the Gaussian similarities between these rare words are becoming richer with the long range knowledge.

Conversely, the best performance for HAL experiments and the cosine kernel are always with the window length of 1 or 2. The more common words, which hold little information gain mass with window length and push the more informative features lower. Table 6.1 (a) demonstrates that at small values of L, HAL matrices with cosine distance hold valid information about word usage. For example, *ptngne1* is similar to *ptngne3* because they both share the context of *ptngne2*.

Although the BEAGLE method does not have window lengths, the distinctions between the Gaussian and cosine similarity transformations hold. The BEAGLE matrix is a rich source of information. That information can be visualised, albeit imperfectly, in reduced dimensional space. Figure 6.6 shows a BEAGLE matrix reduced to three dimensions using principle component analysis (PCA). A small subset of the BC words describing some common biological terms is presented. While many of the words are grouped in together at the origin of the reduced coordinate system, different topics can be distinguished as belonging to the chosen components. One cluster shows protein labels, the more commonly occurring *ptngne1* and *ptngne2* leading the axis away from the origin. Further along this axis, but too far to be effectively represented in this image, is the

feature *gene*, likewise, *bind* and *phosphorylate* belong to the same direction. The second axis going towards the top of the image contains model organisms *yeast, escherichia coli, drosophila, saccharomyces cervisia*, *etc.* In between these axis we see words such as *domain, complex*, and *region*, usually describing the interaction location. Finally the third axis shows words relating to cancer, including a small cluster describing breast cancer: *woman, malignant, mammary* and others such as *tumour, lymphoma, leukemia*. Further along this axis appear words *tumor, apoptosis, inhibitor, treatment*, and *cancer*. Between the "protein" and the "cancer" axis are words such as *vivo, vitro*, and *mice* which are often used in describing experiments relating to examination of interactions in significant pathways. In the large cluster around the origin, there are still distinctions in direction of different words, which can be observed by zooming in. It would appear that the words with higher overall frequency appear further away from the origin, while the ones with the lower frequency are closer. This would indicate that really frequent words, such as gene, have a unique usage, while the model considers other words which appear less frequently, but within similar contexts, more interchangeable.

6.3 Classification experiments

The best way to evaluate these models is to apply them to the feature types that produced the best results in the experiments from the previous chapter. A large set of ten by ten cross-validation experiments searches the space of different types of semantic kernels made from two datasets. The GENIA dataset contains a little over 430 thousand words from abstracts, while the OAA data is a subset of the Open Access full text articles relating to genomics and proteomics and contains over 13 million words. (See Section 2.2.2 for details on corpora.) Both datasets were processed with Lingpipe (Baldwin and Carpenter, 2008), to extract protein names for compatibility with the protein-based feature types. The training data comes from the BioCreative (BC), AImed, and PreBIND (PB) datasets. The MIPS data was discarded, as the classifier performance on abstract data is already quite

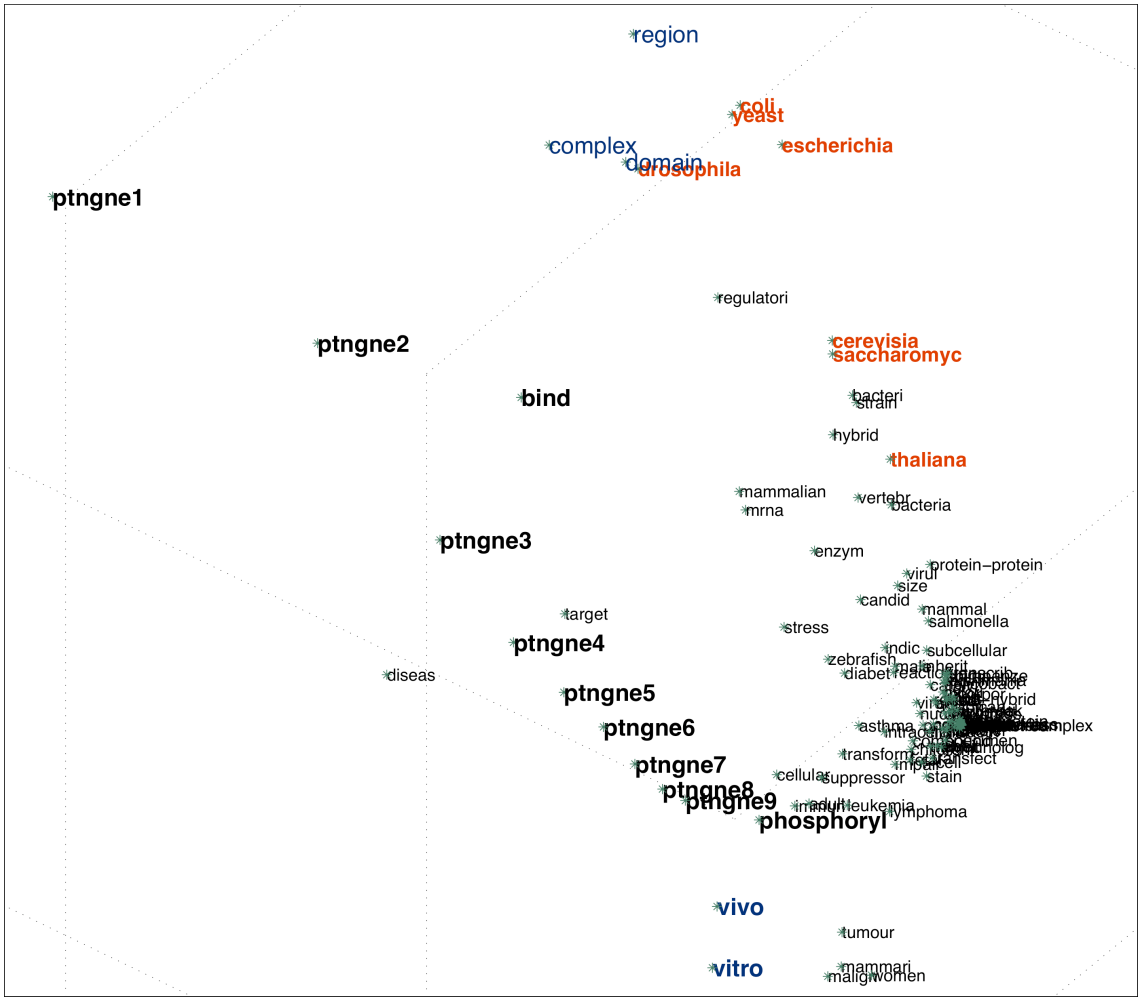


Figure 6.6: Visualisation of the similarities contained in a BEAGLE matrix created from the BC words and the OAA dataset with $D = 2048$.

high, while the automatically annotated sentence data is of low quality. These datasets are also very large with an extensive number of features, which also poses computational difficulties when keeping 15 different matrices in memory. Even for the PB corpus, it was necessary to discard any features occurring only once in the whole dataset.

In the experiment, the unique words from BC, AImed, and PB are transformed into the feature types that produced the best results. These are then used to generate HAL matrices based on different context lengths from both OAA and GENIA. Lengths of $L = 1$ to $L = 15$ were considered. The resulting matrices were converted into T1 and T2 semantic kernels with both cosine and Gaussian similarity functions. For the Gaussian

kernel, several different settings for θ were explored, however, the best results were almost exclusively produced by T1 Gaussian kernels with $\theta = 0.01$. The matrices of the words co-occurring at different lengths within the window are combined using the formula:

$$\mathbf{H_L} = \sum_{l=1}^{L}(L - l + 1)\mathbf{H}_l \tag{6.3}$$

as when constructing the probabilistic HAL kernels. The probabilistic HAL kernels are just normalised HAL kernels ($\mathbf{H}_l$), and this approach is equivalent to normalising the data before creating the semantic kernels. While normalised data improves the classification performance of the T2 kernels, the best results come from using the raw frequency counts both in the labelled data and the in the semantic models.

The BEAGLE kernels all use a sentence as the context, so two different dimension lengths were searched for both OAA and GENIA, $D = 2048$ and $D = 4096$. While this is larger than N for some of the data sets, these matrices were created considering the co-occurrence between the features and up to 30, 000 unique words occurring in the unlabelled data, leading to a substantial reduction in the number of basis. The HAL matrices were constructed by considering the co-occurrence between the features from the training data only.

6.3.1 Results

Table 6.2 shows the numerical results of the cross-validation experiments for both the Gaussian and cosine kernels for all three datasets. As in the previous chapter, the Gaussian kernels show the most stable results and the larger AUC. While the AUC on the experiments using the semantic cosine kernel on the sentence datasets show improvement over the original cosine results, they do not improve over the original Gaussian numbers. However, for the F-score on the AImed data, using either the cosine or Gaussian semantic kernel improves on the best original F-score.

The imbalance in the ratio of positive to negative data samples in the BC corpus leads

BC Protein F4

Original Gaussian	GP $\mathbf{H}_{L=4}$	VBpMKL $\mathbf{H}_{L=12}$	GP $\mathbf{B}_s$	VBpMKL $\mathbf{B}_s$
F=0.4226 ± 0.0361	F=0.4647 ± 0.0117	**F=0.5526 ± 0.0103**	F=0.4636 ± 0.0103	F=0.5405 ± 0.0092
E=14.8369 ± 1.0362	E=14.3959 ± 0.3219	E=13.4632 ± 0.3231	E=14.3644 ± 0.3285	E=13.6945 ± 0.3342
P=0.6757 ± 0.0515	P=0.6469 ± 0.0149	P=0.6548 ± 0.0149	P=0.6581 ± 0.0133	P=0.6528 ± 0.0128
R=0.3235 ± 0.0373	R=0.3764 ± 0.0123	R=0.4943 ± 0.0115	R=0.3690 ± 0.0103	R=0.4739 ± 0.0101
A=0.9227 ± 0.0077	A=0.9226 ± 0.0028	A=0.9224 ± 0.0027	A=0.9214 ± 0.0025	A=0.9203 ± 0.0024

Original Cosine	GP $\mathbf{H}_{L=1}$	VBpMKL $\mathbf{H}_{L=1}$	GP $\mathbf{B}_l$†	VBpMKL $\mathbf{B}_l$†
F=0.5344 ± 0.0094	F=0.2819 ± 0.0116	F=0.4601 ± 0.0121	F=0.0810 ± 0.0078	F=0.2558 ± 0.0175
E=13.6465 ± 0.3243	E=15.1753 ± 0.3363	E=13.8832 ± 0.3285	E=16.7287 ± 0.3586	E=15.4967 ± 0.3717
P=0.6766 ± 0.0135	P=0.7789 ± 0.0219	P=0.7042 ± 0.0170	P=0.5633 ± 0.0478	P=0.6866 ± 0.0310
R=0.4589 ± 0.0115	R=0.1782 ± 0.0084	R=0.3547 ± 0.0119	R=0.0443 ± 0.0045	R=0.1738 ± 0.0140
A=0.8657 ± 0.0047	**A=0.9184 ± 0.0027**	A=0.9128 ± 0.0028	A=0.8950 ± 0.0033	A=0.8854 ± 0.0035

Almed Protein F4

Original Gaussian	GP $\mathbf{H}_{L=3}$	VBpMKL $\mathbf{H}_{L=11}$	GP $\mathbf{B}_s$	VBpMKL $\mathbf{B}_s$
F=0.6712 ± 0.0161	**F=0.7184 ± 0.0044**	F=0.7172 ± 0.0044	F=0.7088 ± 0.0042	F=0.7116 ± 0.0041
E=18.4464 ± 0.8169	E=16.8273 ± 0.2564	E=17.2021 ± 0.2447	E=17.3585 ± 0.2142	E=17.5103 ± 0.2287
P=0.7480 ± 0.0209	P=0.7471 ± 0.0060	P=0.7302 ± 0.0051	P=0.7377 ± 0.0056	P=0.7271 ± 0.0058
R=0.6128 ± 0.0198	R=0.6953 ± 0.0054	R=0.7074 ± 0.0056	R=0.6861 ± 0.0057	R=0.7005 ± 0.0052
A=0.9024 ± 0.0063	**A=0.9052 ± 0.0021**	A=0.9041 ± 0.0020	A=0.9040 ± 0.0018	A=0.9020 ± 0.0019

Original Cosine	GP $\mathbf{H}_{L=1}$ T2	VBpMKL $\mathbf{H}_{L=1}$ T2	GP $\mathbf{B}_l$† T2	VBpMKL $\mathbf{B}_l$ T2
F=0.6875 ± 0.0049	F=0.7227 ± 0.0040	**F=0.7236 ± 0.0042**	F=0.7008 ± 0.0038	F=0.7030 ± 0.0041
E=18.4419 ± 0.2827	E=22.1952 ± 0.2760	E=21.9622 ± 0.2871	E=24.4015 ± 0.2920	E=23.8356 ± 0.2822
P=0.7240 ± 0.0067	P=0.5903 ± 0.0051	P=0.5939 ± 0.0054	P=0.5665 ± 0.0046	P=0.5731 ± 0.0049
R=0.6584 ± 0.0056	R=0.9365 ± 0.0030	R=0.9307 ± 0.0027	R=0.9239 ± 0.0044	R=0.9134 ± 0.0042
A=0.8783 ± 0.0026	**A=0.8997 ± 0.0021**	A=0.8986 ± 0.0021	A=0.8692 ± 0.0026	A=0.8699 ± 0.0025

PB F6

Original Gaussian	GP $\mathbf{H}_{L=6}$†	VBpMKL $\mathbf{H}_{L=6}$†	GP $\mathbf{B}_l$	VBpMKL $\mathbf{B}_l$
F=0.8945 ± 0.0080	F=0.8948 ± 0.0025	F=0.8919 ± 0.0028	F=0.8956 ± 0.0025	**F=0.9026 ± 0.0025**
E=13.5257 ± 0.9660	E=13.3440 ± 0.3009	E=13.4726 ± 0.3308	E=13.3503 ± 0.2942	E=12.3335 ± 0.2959
P=0.8823 ± 0.0112	P=0.8915 ± 0.0036	P=0.9042 ± 0.0034	P=0.8843 ± 0.0039	P=0.9003 ± 0.0038
R=0.9084 ± 0.0107	R=0.8995 ± 0.0036	R=0.8814 ± 0.0041	R=0.9087 ± 0.0032	R=0.9066 ± 0.0034
A=0.9334 ± 0.0079	A=0.9315 ± 0.0023	A=0.9323 ± 0.0024	A=0.9345 ± 0.0020	**A=0.9358 ± 0.0020**

Original Cosine	GP $\mathbf{H}_{L=1}$†	VBpMKL $\mathbf{H}_{L=1}$†	GP $\mathbf{B}_s$†	VBpMKL $\mathbf{B}_l$† T2
F=0.8928 ± 0.0027	F=0.8710 ± 0.0032	F=0.8694 ± 0.0031	F=0.8556 ± 0.0031	F=0.8080 ± 0.0033
E=13.7267 ± 0.3207	E=16.6781 ± 0.3728	E=16.4575 ± 0.3556	E=20.0716 ± 0.3889	E=21.6489 ± 0.3751
P=0.8807 ± 0.0036	P=0.8503 ± 0.0042	P=0.8697 ± 0.0040	P=0.7850 ± 0.0051	P=0.9237 ± 0.0039
R=0.9066 ± 0.0035	R=0.8945 ± 0.0040	R=0.8706 ± 0.0040	R=0.9428 ± 0.0027	R=0.7200 ± 0.0047
A=0.9302 ± 0.0025	A=0.9093 ± 0.0027	A=0.9099 ± 0.0027	A=0.8998 ± 0.0031	A=0.8823 ± 0.0033

Table 6.2: Best results from the semantic kernel experiments. The † symbol indicates that the GENIA dataset lead to the best results otherwise it was OAA. Likewise if T2 is not specified, the results were obtained using the T1 semantic kernel.

to low F-scores, which are adversely affected by the semantic cosine kernels, despite the improvement in the AUC. This indicates that, although the positive documents have a higher ranking than the negative ones, the default cutoff probability of 0.5 is too high. The ratio of precision and recall can be changed by varying the cutoff threshold. For this reason the AUC is considered a more accurate representation of the algorithm performance. Apart from this one case, a small improvement in the AUC generally produces a

larger improvement in the F-score, as can be visualised in Figures 6.7, 6.8, and 6.9.

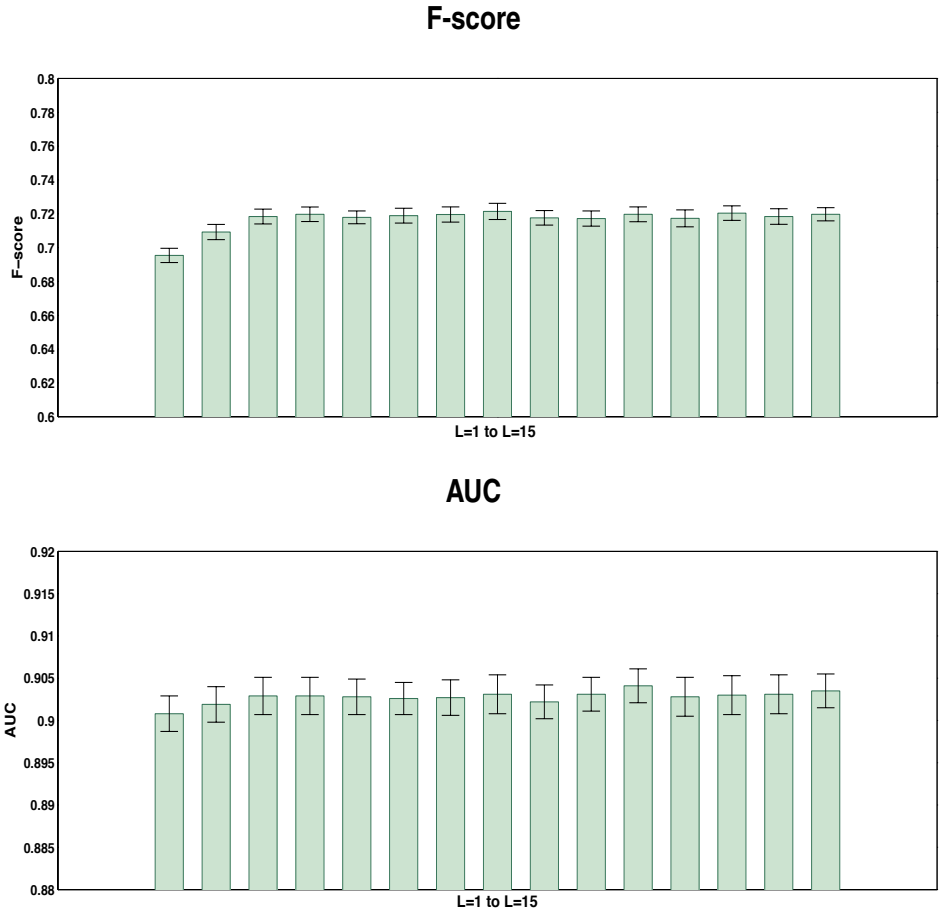


Figure 6.7: Classification performance of HAL-based Gaussian semantic kernels created from different context lengths (BC).

It is also obvious from these figures that Gaussian HAL matrices of different lengths produce results with very little variation; however, there is a slight improvement in the AUC as the context window length, L, increases. The opposite is true for the cosine HAL matrices, whose performance degrades with the increase L.

As Figure 6.9 shows GENIA and OAA semantic kernels produce different results. In general, the semantic kernels created from the larger OAA corpus produces higher classification scores. The exceptions can be seen with the PB dataset and when using certain BEAGLE cosine kernels; here, using the similarities learned from the GENIA corpus results in the higher AUCs. Both GP and VBpMKL get similar AUC and F-scores,

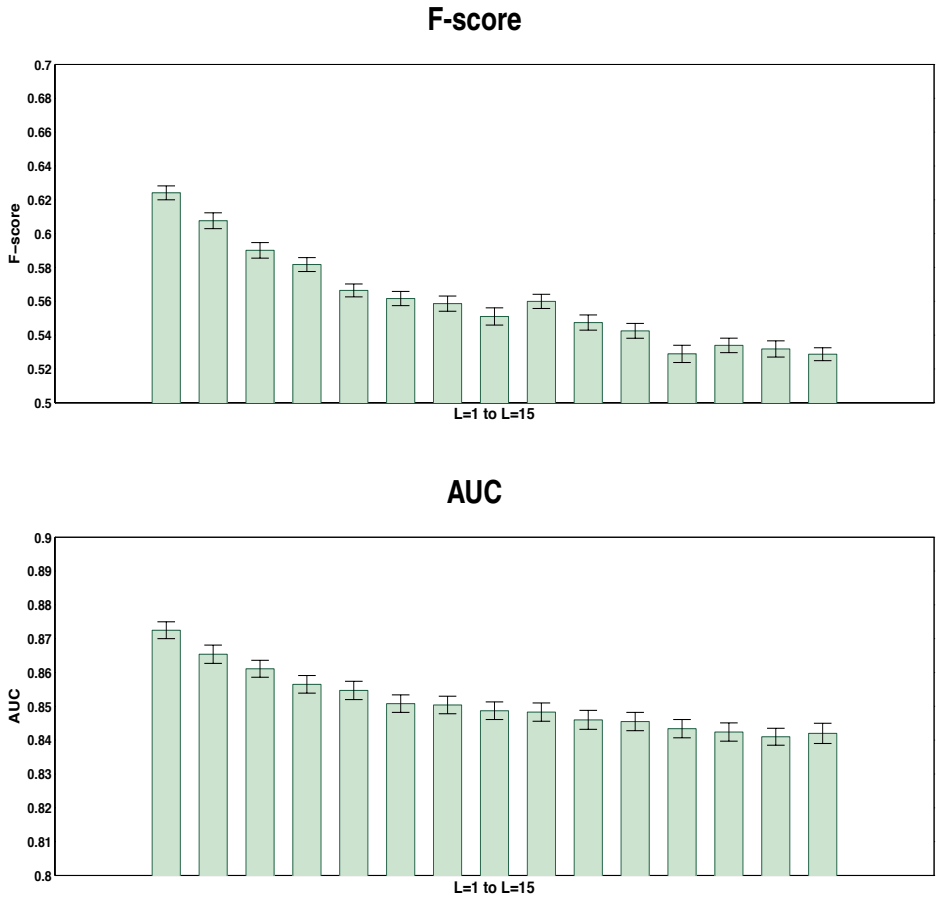


Figure 6.8: Classification performance of HAL-based cosine semantic kernels created from different context lengths (BC).

but not necessarily with the same kernels.

6.4 Latent Dirichlet allocation on the Almed $\mathbf{H}_{L=3}$ matrix

LDA is most often used to determine the distribution of topics that make up a document as a way of doing soft clustering. In this section, however, LDA is used to examine the quality of the semantic information contained in a HAL matrix. LDA assigns several topics to a single word, in this case based on its immediate context, as opposed to the long documents in which it occurs. Such clustering provides a more structured visualisation of word similarity than, for example, lists ordered by cosine distance between words. It also provides an opportunity to assess ability of LDA as a method for reducing the

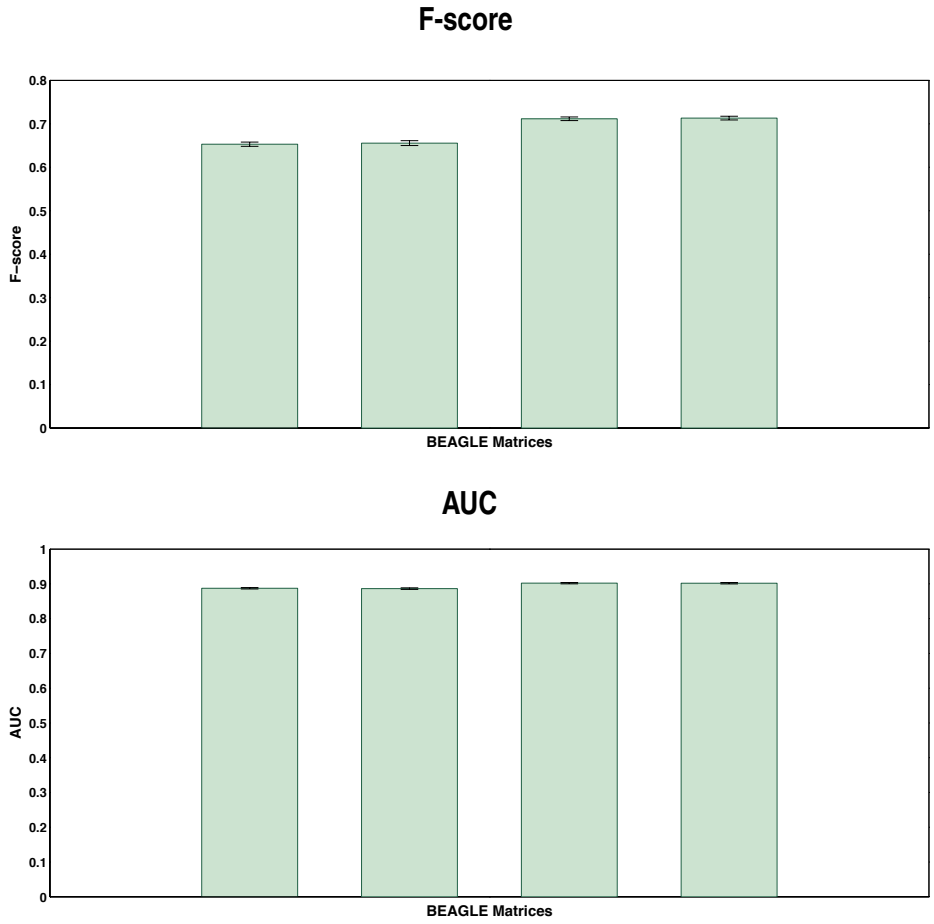


Figure 6.9: Classification performance of BEAGLE-based Gaussian semantic kernels created from different datasets and with different Ds (BC data). The first two kernels are created from GENIA while the last two are from OAA. The first and third kernel have $D = 2048$, while the second and fourth have $D = 4096$.

dimensionality of the $N \times N$ HAL co-occurrence matrix.

Table 6.3 shows two sets of example topics from LDA models trained on the AImed sentence-feature matrices. One set comes from an LDA model trained to detect 40 topics, and the other 400 topics; however, the subjects of all of these example topics are difficult to identify. Training the classifiers in this reduced-dimensional space also produces an inferior model. With forty or four hundred topics the classification AUC drops nearly 10%. Therefore, the AImed dataset, which contains under 2,000 sentences, is too small for training of an accurate topic model.

In the HAL matrices used for semantic kernels, the usage of the words occurring in the

LDA on AImed with 40 Topics

Topic 1	Topic 2	Topic 3	Topic 4
pulldown	pulldown	test	tata
ptngne5	experiment	pulldown	orient
modul	ptngne5	immunosuppress	intrins
respons	marker	make	nonallerg
trimer	nonallerg	s252w	attach
ptngne2	modul	subset	pulldown
destroi	electrophoret	biogen	ptngne5
cultur	scarc	tata	background
ants2	steroid	import	glu
gel	vocal	treat	dimension

LDA on AImed with 400 Topics

Topic 1	Topic 2	Topic 3	Topic 4
salt	malnourish	nonallerg	ptngne2
proteolysi	depend	inhibit	ptngne5
ly	uroepitheli	select	ptngne7
begin	glycin	sequenc	pulldown
onc	oviduct	anomal	experiment
snyder	brucei	respect	ber
aneurysm	glutam	multival	nonallerg
purifi	predominantli	appropri	tata
belgium	intrins	multitud	modul
gitr	divers	site	subset

Table 6.3: The figure shows results from LDA trained on AImed documents. The sentences in the corpus were considered documents, while the words were considered the features. The unique topics in the figure demonstrate the inferred semantic groupings extracted by the LDA algorithm seeded with the presumed number of topics. The top group shows results if the assumed number of topics is 40 and the bottom shows the results when it is 400.

AImed data is tracked across a much larger dataset. The words are used in many different contexts and some of which reflect the usage of these words in the original sentence data. While the documents in training data were the sentence vectors, in HAL matrices these documents are all possible contexts a word can occur in, within a specified window. Table 6.2 shows that for the AImed dataset the highest improvement in classification is found when the $\mathbf{H}_3$ matrix created on the OAA dataset is used in conjunction with the GP classifier. $\mathbf{H}_3$ is the HAL matrix containing the number of times each target word co-occurs with any of the basis, within a window of three words to the either side of it. Each target vector is considered a document in the context of LDA.

There is a variation in the LDA likelihood with an increase in the number of topics. The initially steady improvement becomes unstable after 80 topics with the highest spike at 180 topics (Figure 6.10). Tables 6.4 and 6.5 show samples of topics from LDA on $\mathbf{H}_3$

trained with 40 and 400 topics, respectively. Initial inspection of topics shows increased coherence compared to the topics trained on the AImed sentence data.

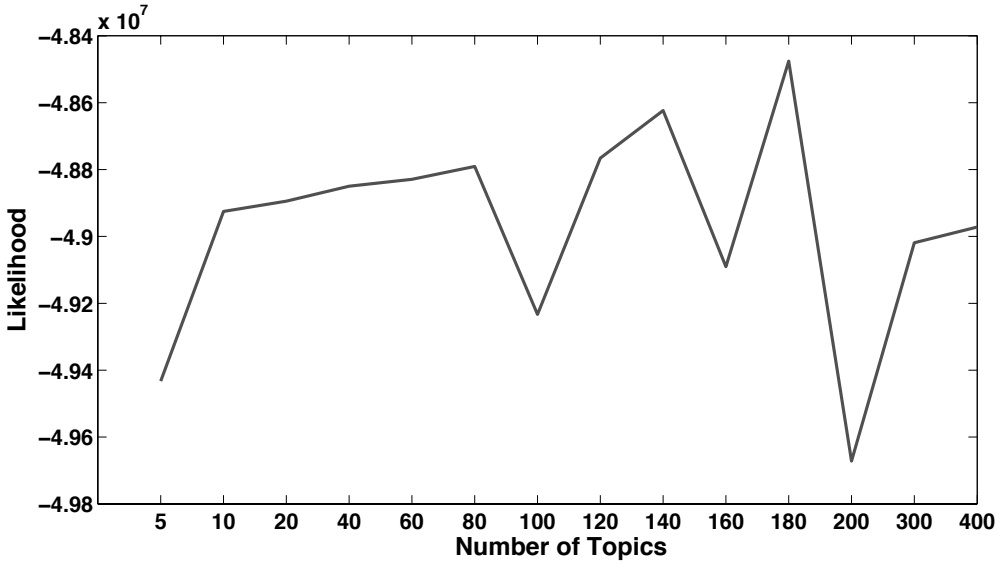


Figure 6.10: Likelihoods for the different number of topics of the LDA on the $\mathbf{H}_3$ matrix.

Observation of the words belonging to each of the topics shows an improvement of topic quality with the increase of number of topics. Even with 40 topics there is a sensible grouping of terms. Table 6.4 shows eight topics, that are thematically vertically aligned, to demonstrate how similar words can appear in different contexts. Topics 1 and 5 show how the word *sequence* has different topical connotations depending on whether it is used to describe protein or gene sequences. Many of the topics extracted from this matrix, which describes how AImed words are used in the OAA dataset, are headed by the word *cancer*. This is an area of medical research that is subdivided into many different topics reflecting the types of diseases that are being studied and approaches taken. Topic 2 describes studies of breast cancer in tissue samples while topic 6 describes treatment of live patients. Topics 3 and 7 contrast cell population studies conducted in a biomedical laboratory versus the data modelling studies conducted on a computer. Finally, topics 4 and 8 show different views of inter-cellular activities, with one topic describing cell proliferation studies with regards to cancer, while the other describes cell signalling and protein pathways.

Eight more, this time unaligned, topics are shown in Table 6.5. These samples come from the LDA trained with 400 topics. Topic 1 describes human gene expression experiments, and interestingly hints that sentences with three or four proteins may describe protein-gene interactions that occur during transcription. The rest of the topics show even more coherence within the subject of the top words that are included. This can be shown by contrasting the breast cancer topic in the two tables. Topic 2 in Table 6.4 corresponds to Topic 5 in Table 6.5, but the words in the latter appear to be more consistent with the listing of both *tumor* and the alternative spelling *tumour*, as well as the relevant species, *i.e. human*.

Topic 1: gene sequence	**Topic 2: breast cancer**	**Topic 3: cell population studies**	**Topic 4: cell proliferation**
gene	cancer	popul	cell
sequenc	breast	differ	activ
genom	cell	type	receptor
dna	human	studi	express
region	gene	group	factor
human	tumor	cancer	growth
chromosom	line	rate	inhibit
clone	carcinoma	cell	effect
pcr	ptngne1	between	induc
express	tissu	control	tumor

Topic 5: protein sequences	**Topic 6: cancer patient study**	**Topic 7: data modelling studies**	**Topic 8: cell signalling**
sequenc	cancer	model	cell
align	patient	method	regul
protein	treatment	data	signal
analysi	studi	analysi	protein
genom	trial	estim	activ
multipl	therapi	test	pathwai
structur	breast	gene	kinas
method	advanc	rate	role
predict	phase	effect	develop
gene	express	statist	transcript

Table 6.4: Eight topics sampled from the LDA trained on $\mathbf{H}_3$ with 40 topics. Pairs of similar topics are aligned vertically. For example, topics 1 and 4 talk about sequences, while 1 contains words more concerned about gene sequences, 4 talks more about proteins their structure and sequence alignment.

The quality of the topics is so high that using the HAL space reduced to these 40 topics to create a semantic kernel produces a GP classification AUC of 0.9037 ± 0.0020. This is already an improvement over the original Gaussian kernel results. Using a HAL matrix reduced to 180 topics produces AUC of 0.9055 ± 0.0022, a value slightly higher than the one listed in Table 6.2 for the results with full $\mathbf{H}_3$. Finally, semantic kernel created from

Topic 1: human gene	Topic 2: E.coli sequencing	Topic 3: cancer drug trial	Topic 4: coronary disease
gene	coli	trial	blood
cell	gene	cancer	cholesterol
ptngne3	escherichia	patient	diseas
express	protein	studi	gene
human	express	random	lipoprotein
ptngne4	regul	treatment	heart
activ	sequenc	therapi	famili
sequenc	genom	result	apolipoprotein
type	transcript	effect	studi
studi	analysi	phase	risk

Topic 5: breast cancer	Topic 6: statistical analysis	Topic 7: DNA replication	Topic 8: neural cells
breast	data	dna	neuron
cancer	statist	cell	brain
carcinoma	analysi	protein	receptor
express	sequenc	replic	cell
cell	gene	genom	protein
tumor	ptngne1	gene	rat
tumour	perform	recombin	express
human	version	repair	mice
invas	signific	coli	system
tissu	studi	yeast	gene

Table 6.5: A sample of eight topics from the LDA trained on $\mathbf{H}_3$ with 400 topics.

the $\mathbf{H}_3$ reduced to 400 topics gives an AUC of 0.9041 ± 0.0024.

In conclusion, the LDA topics that are collected from a smaller context window on a larger corpus are of visibly higher quality than the ones directly collected from the original training data. These topics can also be used to reduce the size of the HAL matrices, although this method of dimension reduction is more computationally intensive than the random indexing employed in BEAGLE. The reduced representation is as effective for use in semantic kernels as the full HAL matrices. The LDA likelihood could be a good indicator of how many topics are optimal for a reduced representation of the matrix, for classification purposes; however, a larger number of topics produces visibly better word groupings and also seems to lead to higher classification AUC.

6.5 Discussion

Semantic kernels provide a smoothing of the training data by altering the weights of the words according to their usage in general. The lexical co-occurrence models trained on a large, relevant subset of freely available open access articles, in general, produced better

classification results. The semantic kernel approach described here is similar to the methods using semantic kernels created from Word Net (Fellbaum et al., 1998) or Wikipedia information (Basili et al., 2005; Minier et al., 2007); however, manually constructed ontological lexical data, such as this, is not available for biomedical words. Using automatically derived semantic information for classification allows us to overcome this lack of data. It also gives us a way of evaluating the quality of word co-occurrence matrices, which is a difficult task usually requiring specialised human judgements.

The quality of word co-occurrence matrices can also be assessed visually by inspecting the groupings of the data in a space whose dimensions have been reduced by a latent topic model such as LDA or PCA. PCA allows graphical visualisation of data in 2D or 3D space, while LDA shows the words sorted by latent topics with which they are associated. The HAL matrices are particularly suited for LDA, and the topics are determined by the contexts within which the words frequently occur. Words with shared contexts are grouped in similar topics. The quality of the HAL-based LDA model is visibly higher than the quality of the document based LDA model gathered from the original data. The reduced dimensional representation produced by the LDA can also be used to generate high-quality semantic kernels.

In general, using the semantic kernels produced an improvement in either the F-score or AUC or both. For BC and UT the improvement in the F-score was statistically significant ($p = 0.0011$ and $p = 2.16e^{-9}$ respectively), while the AUC was not, due to high variance in the original experiments. For the PB data, only the BEAGLE semantic Gaussian kernel produced improvement. This might be due to the length of document, for example Song and Bruza (2001) found that using HAL to expand longer queries was not as effective as it was for shorter queries. Abstracts contain sentences which do describe PPIs as well as ones that do not.

In this chapter contributions of individual semantic kernels were assessed. In the following chapter the results of these experiments will be combined to investigate the use of VBpMKL for learning from multiple semantic kernels.

Chapter 7

Semantic Kernel Combination

In Chapter 6 we showed that using semi-supervised learning through semantic kernels on protein-protein interaction sentence data can boost the AUC and the F-score over the best supervised results. While this increase was larger for the F-score, it was not overall statistically significant. In this chapter we will further improve on those results by using combinations of the best kernels to train the multiple kernel learning classifier VBpMKL.

Kernel combination algorithms allow us to build one classifier that learns from multiple feature spaces. This is contrasted with ensemble learning, where each feature space results in a new classifier. The results of these separate classifiers are then polled to produce the final classification decisions. In multiple kernel learning, the different feature spaces are first transformed into kernels and then combined into a single kernel, which can be used in a kernel method such as the GP, SVM, or VBpMKL classifier.

The VBpMKL algorithm, used in this chapter, can estimate the weights that govern the contribution of each individual kernel to the final combined kernel. The VBpMKL algorithm and the kernel combination methods were introduced in Section 3.3.5. Two methods for combining kernels are explored in this thesis:

- the *uniform* combination gives each kernel an equal weighting of 1 and the sum remains unnormalised;
- the *convex linear* method learns the weights, which maximise the classification

model likelihood, through sampling.

The weights of each contributing kernel are denoted by β_s. In the uniform approach, these weights sum up to the number of kernels, $\sum_{s=1}^{S} 1 = S$. This gives equal and unnormalised weight to each kernel, thus widening the range of similarity between the points. Points judged as highly similar by more of the kernels will increase closer to S, while the points that do not bear consensus will remain lower. In the convex linear combination, on the other hand, the weights (β_s) of the kernels are bound to sum to 1, $\sum_{s=1}^{S} \beta_s = 1$, and thus are operating on a different scale from the uniform sum. The estimated values of β_s indicate the informativeness of a kernel. If two kernels contribute similar information one will be weighted higher than the other, and thus will reduce the compounding effect produced by uniform summation of kernels.

The next section describes the aims of the experiments and the experimental setup. It is followed by a section presenting the results and finally a discussion of the observations from this chapter.

7.1 Kernel combination experiments

The experiments in this chapter are divided into two sections, the first describes the experiments and results using the uniform kernel weightings, while the second covers the convex linear approach and the experimental results.

Apart from this difference the construction of the experimental methods is the same. This structure is laid out in Figure 7.1. When reading the diagram from left to right we can see how the unlabelled data is combined with labelled data into T1 semantic kernels, in the manner described in Chapter 6. In this chapter the labelled data comes from the AImed and BC sentence corpora. These two datasets are smaller than the PB data in the previous corpus, and thus allow for easier computation because the wide range of experiments performed in this chapter sometimes requires many kernels to be held in memory at once. The semantic kernel results, from Chapter 6, showed that the BC dataset

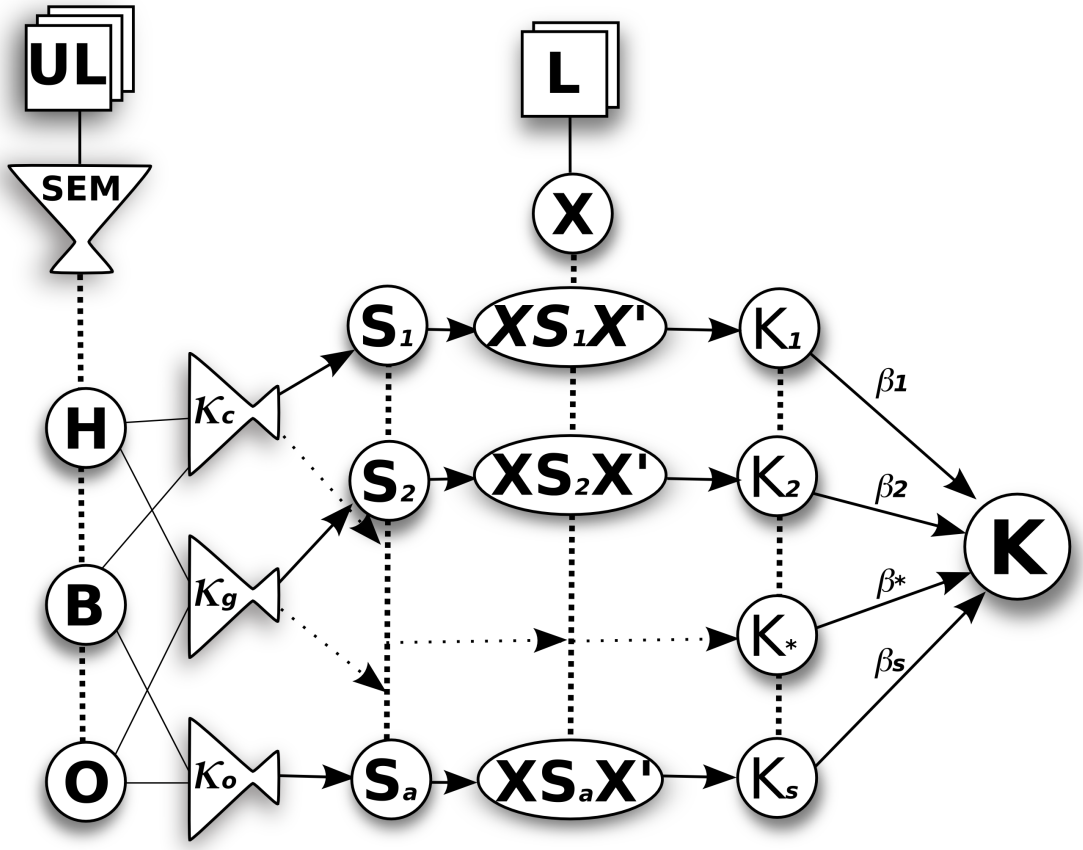


Figure 7.1: The overview of the method. The training data ($\mathbf{X}$) comes from the labelled corpus ($\mathbf{L}$), while the unlabelled data ($\mathbf{UL}$) is transformed using semantic models ($\mathbf{SEM}$) to produce word co-occurrence matrices, such as $\mathbf{H}$, $\mathbf{B}$, or other $\mathbf{O}$. These matrices are then passed to one or more of the available similarity metrics, such as the cosine (κ_c), Gaussian (κ_g), or other kernel functions (κ_o). The resulting similarity smoothing matrices are combined with the training data $\mathbf{X}$ to produce semantic kernels which are then combined into a single kernel ($\mathbf{K}$), with weightings β_s.

AUC only improved on the cosine kernel, while the AImed AUC improved on both the cosine and Gaussian kernels. The goal of these experiments will be to combine the best performing kernels in such a way as to improve upon the best results from the previous chapter.

The unlabelled data from the GENIA and OAA datasets is processed by the unsupervised BEAGLE and HAL algorithms to extract the co-occurrence matrices. These matrices are converted to word-word distance matrices by a kernel function. This thesis continues to use the cosine and Gaussian distance metrics, although any other kernel

function can also be applied.

Finally, the resulting kernels are combined into a single kernel using a fixed or an estimated weighting technique.

7.1.1 Combinations of HAL kernels

There are two experiments specifically aimed at exploring HAL semantic kernels. In the first experiment $\mathbf{H}_{15}$ is decomposed into its constituents and different weighting schemes are examined. In the second experiment the contributions of each of the composite kernels $\mathbf{H}_L$ are evaluated.

The following are the detailed descriptions of the experiments:

1. HAL matrices $\mathbf{H}_L$ are already a weighted combination of matrices, $\mathbf{H}_l$, containing the co-occurrence counts between targets and basis occurring at l words before or after the target, where $1 < l < L$. These kernels are combined using a weighting function which boosts the importance of matrices that represent co-occurrence with words closer to the target:

$$\mathbf{H}_L = \sum_{l=1}^{L} (L - l + 1)\mathbf{H}_l$$

   By turning each of the matrices $\mathbf{H}_l$ into an individual semantic kernel and combining them using the uniform weighting, words at each of the distances 1 to 15 away from the target are given the same priority:

$$K = \sum_{l=1}^{15} \mathbf{X}\kappa(\mathbf{H}_l, \mathbf{H}_l)\mathbf{X}^T$$

   To explore the importance of the contributions of each of the composing matrices, $\mathbf{H}_l$, this same experiment is also repeated using the learned weighting for each of the matrices.

2. In the second experiment the uniform weighting is used to see if a combination of the different $\mathbf{H}_L$ matrices:

$$K = \sum_{L=1}^{10} \mathbf{X}\kappa(\mathbf{H}_L, \mathbf{H}_L)\mathbf{X}^T$$

will improve on the classification results of the best single matrix $\mathbf{H}_{11}$. Similarly, the convex linear weighting is used to learn the contributions of the different matrices and to see how well they correlate with the results on each of the individual matrices from the previous chapter.

In both of these experiments, HAL was trained on the OAA dataset, because the results from the previous chapter indicate that these matrices lead to a better performance on both the AImed and BC corpora. The Gaussian kernel is applied for the same reason.

7.1.2 Combinations of BEAGLE kernels

BEAGLE-based kernels do not have as many options to consider as the HAL-based ones. The main parameter that needs exploration is the number of dimensions chosen for the environmental vectors that encode each word. In the previous chapter the experiments concentrated on $D = 2048$ and $D = 4096$ dimensions, referred to as BEAGLE small, $\mathbf{B}_s$, and BEAGLE big, $\mathbf{B}_b$, respectively. Both BC and AImed results indicated best performance with the $\mathbf{B}_s$ matrix trained on the OAA dataset; however, Figure 6.9 shows that the preference for this matrix is not overwhelming. Thus in this chapter, the VBpMKL is trained on a combination of $\mathbf{B}_s$ and $\mathbf{B}_b$ trained on both GENIA and OAA. Similarly to the HAL experiments above, their contributions are evaluated using the learned convex linear weighting.

7.1.3 Combinations of best-performing kernels

Finally, after comparing the best results from the above with the results from the previous chapter, the best kernels are combined to try to produce further improvements.

7.2 Results

This section contains results from the experiments on different combinations semantic kernels. These are evaluated against the best results from the Chapters 5 and 6, which are repeated for comparison against the best AUC and F-scores from this section in Table 7.4.

7.2.1 Context length of HAL matrices

These experiments compare the uniform and learned sums of $\mathbf{H}_l$ matrices with the $L-l+1$ weighted sum that is usually used to combine HAL matrices. Table 7.1 compares the $\mathbf{H}_{15}$ single kernel result for VBpMKL with the uniform and convex linear sums of kernels constructed from $\mathbf{H}_l$ matrices for $l = 1$ to $l = 15$.

BC Protein F4

$\mathbf{H}_{15}$	Uniform	Convex Linear
F=0.5458 $\pm$ 0.0102	**F=0.6165 $\pm$ 0.0088**	F=0.4249 $\pm$ 0.0120
E=13.5931 $\pm$ 0.3130	E=12.1632 $\pm$ 0.3243	E=14.9564 $\pm$ 0.3175
P=0.6468 $\pm$ 0.0125	P=0.6799 $\pm$ 0.0109	P=0.6352 $\pm$ 0.0163
R=0.4898 $\pm$ 0.0126	R=0.5762 $\pm$ 0.0113	R=0.3345 $\pm$ 0.0119
A=0.9210 $\pm$ 0.0026	**A=0.9288 $\pm$ 0.0027**	A=0.9126 $\pm$ 0.0026

Almed Protein F4

$\mathbf{H}_{15}$	Uniform	Convex Linear
F=0.7128 $\pm$ 0.0038	**F=0.7216 $\pm$ 0.0044**	F=0.6808 $\pm$ 0.0088
E=17.4793 $\pm$ 0.2355	E=17.2270 $\pm$ 0.2360	E=18.2586 $\pm$ 0.3242
P=0.7274 $\pm$ 0.0048	P=0.7223 $\pm$ 0.0052	P=0.7393 $\pm$ 0.0063
R=0.7016 $\pm$ 0.0053	R=0.7243 $\pm$ 0.0060	R=0.6438 $\pm$ 0.0103
A=0.9035 $\pm$ 0.0020	A=0.8982 $\pm$ 0.0021	A=0.8975 $\pm$ 0.0025

Table 7.1: The results of uniform and convex linear combinations of HAL kernels created of individual context lengths.

For the BC data, the uniformly-weighted combination kernel produces the highest AUC and F-scores, higher even than the best scores from the experiments from the previous chapters. For the AImed data the AUC is highest with the single kernel, while the F-score is slightly higher with the uniform combination. The convex linear combination for both data sets is lower.

The VBpMKL kernel weights can be used to assess the amount of information found at a certain distance from the target. Figures 7.4 and 7.5 show how these weights vary depending on the dataset. High β values indicate that the corresponding kernels produced an increase in the VBpMKL model likelihood. These values indicate that for both datasets the window of words close to the target is highly informative, supporting the ramped weighting employed in the HAL approach. For the AImed data, however, matrices containing words co-occurring at the distance of 11 to 15 are given even higher weights. In the single kernel experiments, highest AUC values for both datasets were achieved when the $\mathbf{H}_L$ matrices were constructed with longer context distances. These results indicate that, for the BC data, the learned weights underestimate the contribution of basis words further away from the targets, while in the AImed their contributions are overestimated.

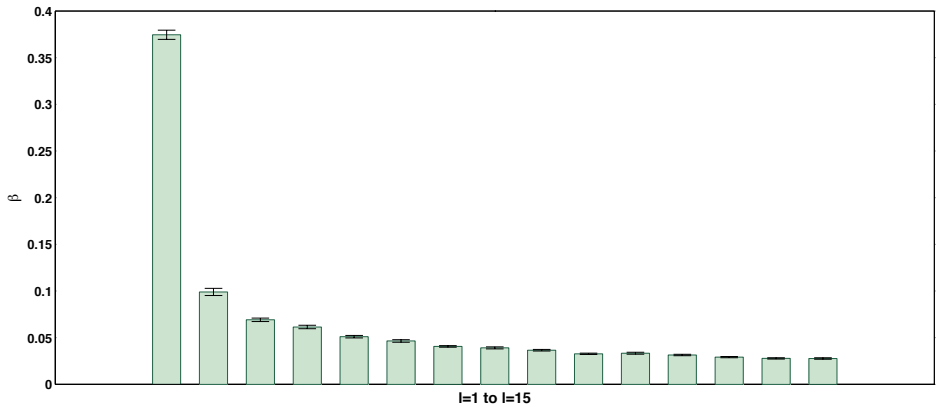


Figure 7.2: Betas for the individual window legths l of the HAL kernels ($\mathbf{H}_l$) for BC data.

7.2.2 Amount of information in each of the HAL matrices

In the previous set of experiments we saw a decrease in AUC when the kernel weights were estimated; however, the convex linear algorithm provides better results than the simple uniform sum when all of the $\mathbf{H}_L$ matrices are combined into a single kernel. Table 7.2 shows that for BC this performance is also better than the best results from the previous chapter, but worse than the uniform combination of the $\mathbf{H}_l$ matrices from the previous section.

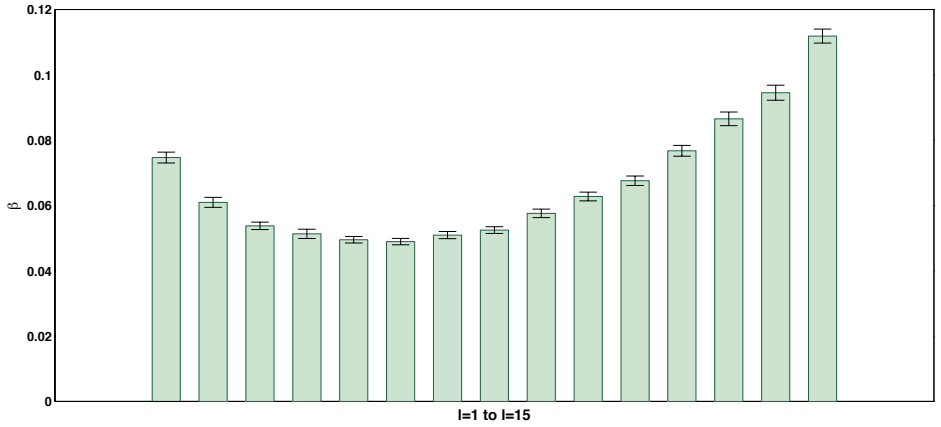


Figure 7.3: Betas for the individual window legths l of the HAL kernels ($\mathbf{H}_l$) for AImed data.

There is a large information overlap between the $\mathbf{H}_L$ matrices. The $\mathbf{H}_{L=1}$ matrix contains only the AImed words that are closest to each other, and with the weight 1, in essence equal to $\mathbf{H}_{l=1}$. On the other hand, the $\mathbf{H}_{L=15}$ matrix also contains these words with weight 15, in addition to all the other words that occur within 15 relevant words of the target, with decreasing weighting based on distance.

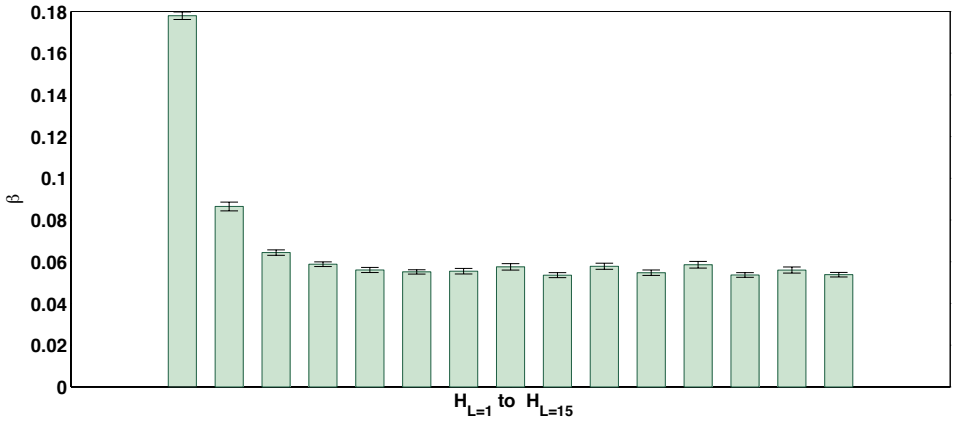


Figure 7.4: Kernel weights (β) for the HAL kernels ($\mathbf{H}_L$) for BC data.

Figures 7.4 and 7.5 show the weightings that were assigned by the algorithm. Figure 7.4 bears a striking similarity to Figure 6.8; however, Figure 6.8 describes the AUC of single kernel experiments on the BC dataset with the cosine semantic kernels. The experiments in this chapter were only conducted with the Gaussian kernel. The best per-

BC Protein F4

Uniform	Convex Linear
F=0.6326 $\pm$ 0.0099	F=0.5209 $\pm$ 0.0117
E=12.1339 $\pm$ 0.3148	E=13.6928 $\pm$ 0.3641
P=0.6625 $\pm$ 0.0125	P=0.6575 $\pm$ 0.0137
R=0.6230 $\pm$ 0.0127	R=0.4464 $\pm$ 0.0130
A=0.9134 $\pm$ 0.0038	A=0.9251 $\pm$ 0.0027

Almed Protein F4

Uniform	Convex Linear
F=0.7080 $\pm$ 0.0043	F=0.6884 $\pm$ 0.0069
E=18.4865 $\pm$ 0.2735	E=17.5912 $\pm$ 0.2975
P=0.6950 $\pm$ 0.0058	P=0.7631 $\pm$ 0.0067
R=0.7253 $\pm$ 0.0053	R=0.6401 $\pm$ 0.0106
A=0.8816 $\pm$ 0.0024	A=0.9019 $\pm$ 0.0026

Table 7.2: The results of uniform and convex linear combinations of all HAL kernels created with $\mathbf{H}_L = \sum_{l=1}^{L}(L-l+1)\mathbf{H}_l$.

formance for the Gaussian semantic kernel come from the $\mathbf{H}_{11} = 11\mathbf{H}_{l=1} + 10\mathbf{H}_{l=2} + \cdots + \mathbf{H}_{l=11}$. Thus the importance of $\mathbf{H}_{l=1}$ is reflected in the weighted combined kernel construction and the words closest to the target contribute the most information. This information is duplicated in each kernel and thus reinforced. The higher weight in the distant kernels further reinforces the similarities from $\mathbf{H}_{l=1}$, while showing the importance of some of the words which are further away from the target. Figure 7.5 supports the hypothesis postulated by the single kernel results, that for AImed data, some information pertinent to the classifier is contained in distant relationships.

7.2.3 Combinations of BEAGLE kernels

Experiments in Chapter 6 showed that various BEAGLE matrices all lead to a high classification AUC regardless of the number of dimensions. There was likewise only a small difference in the AUC depending on the BEAGLE training data, with the OAA dataset providing slightly better information. The uniform combination of all of these BEAGLE-based semantic kernels provides the highest AUC and F-score results, so far.

Table 7.3 compares cross-validation results of the classifiers trained on the uniform

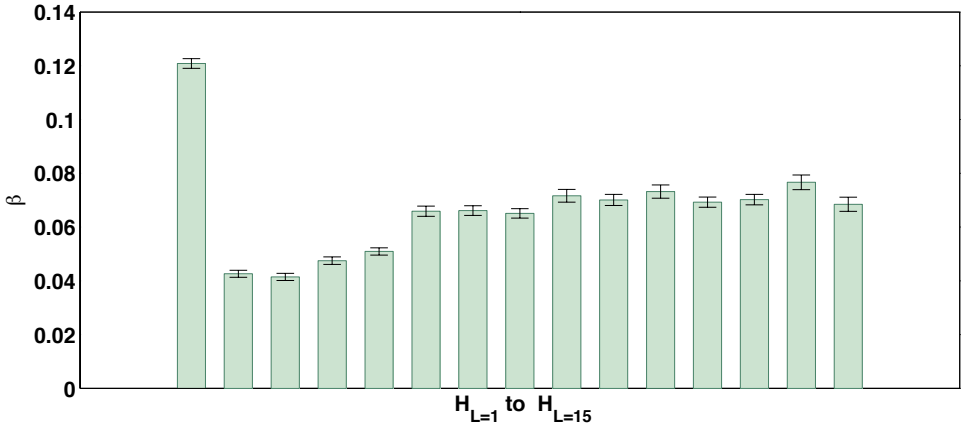


Figure 7.5: Kernel weights (β) for the HAL kernels ($\mathbf{H}_L$) for AImed data.

and convex linear combinations of all BEAGLE-based kernels. The BC uniform scores are over 1% higher than the baseline results, while the AImed ones are nearly over 0.5% higher.

The convex linear method for both datasets does not perform as well as the uniformly combined kernel, but the AImed AUC is almost equal to the best single kernel results. Figures 7.6 and 7.7 show the weights assigned to each of the BEAGLE-based semantic kernels. While it would initially appear that the algorithm is preferring the BEAGLE matrices only based on the generating unlabelled data, closer examination shows that it is also detecting the duplication of information between the kernels made from BEAGLE matrices with the same dimensions. In fact, kernels 1 and 3 have been created from BEAGLE matrices with the dimensions $N \times 2048$, while kernels 2 and 4 come from $N \times 4096$ BEAGLE matrices. The weights are split at almost exactly half by the dimensions, that is for the BC data $\beta_1 + \beta_3 = 0.5021$ whilst $\beta_2 + \beta_4 = 0.4979$. Similarly for the AImed data, we have $\beta_1 + \beta_3 = 0.5012$ and $\beta_2 + \beta_4 = 0.4988$.

Thus the method is sensitive to small variations in information contributions due to the difference in the dimensions used to create the original semantic matrices. By choosing the kernels created from the OAA data, the convex linear scores are equivalent to using the single best kernel. On the other hand, choosing the non-optimal data for the BC corpus causes lower AUC and F-score.

BC Protein F4

Uniform	Convex Linear
F=0.6217 ± 0.0105	F=0.5094 ± 0.0102
E=11.6227 ± 0.3225	E=14.0352 ± 0.3622
P=0.7048 ± 0.0114	P=0.6552 ± 0.0146
R=0.5720 ± 0.0131	R=0.4316 ± 0.0109
A=0.9339 ± 0.0024	A=0.9168 ± 0.0028

AImed Protein F4

Uniform	Convex Linear
F=0.7414 ± 0.0039	F=0.7048 ± 0.0073
E=15.9214 ± 0.2291	E=17.3179 ± 0.3007
P=0.7474 ± 0.0056	P=0.7442 ± 0.0064
R=0.7395 ± 0.0054	R=0.6820 ± 0.0101
A=0.9105 ± 0.0019	A=0.9051 ± 0.0023

Table 7.3: The results of uniform and convex linear combinations of all the BEAGLE kernels.

7.2.4 Engineering the best kernel combination

The previous sections provided excellent improvements on the baseline results, when using uniform combinations. In Chapter 6, the highest single kernel results were achieved by using the kernels constructed using HAL matrices trained on the OAA data and transformed by the Gaussian kernel. The highest AUC from the BEAGLE-based semantic kernels came from the $\mathbf{B}_s$ matrix trained on the OAA data. Combining these for BC improves upon the baseline, but not over the results achieved in Section 7.2.3. For AImed the combined AUC is lower than for either the best HAL or BEAGLE single kernel scores.

In this chapter the highest AUC and F-score for both AImed and BC data was produced by the uniform combination of all BEAGLE kernels. Adding the best single HAL-based kernel to this combined kernel, further increases the AImed AUC, but slightly lowers the BC AUC. Table 7.4 shows the best results from this chapter and compares them with the highest results from Chapters 5 and 6. There is a 10% improvement over the F-scores in the BC data along with the highest AUC from all the different feature and kernel types tested so far. A similar increase can be seen in the AImed data where the F-score is 7% higher than the best possible from the original data, along with a statistically signifi-

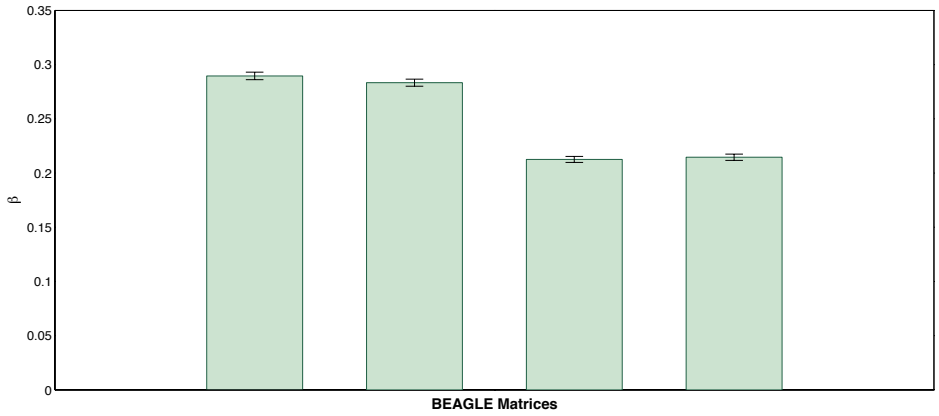


Figure 7.6: Betas for different BC BEAGLE T1 semantic kernels. The first two kernels are created from GENIA while the last two are from OAA. The first and third kernel have $D = 2048$, while the second and fourth have $D = 4096$.

cant increase in the AUC. The AUC is a harder measure and small gains usually translate into larger gains in the F-score, although the two are not necessarily always linked, as was seen in semantic cosine kernel results from the previous chapter.

7.3 Discussion

Kernel combination methods enable simultaneous learning from multiple feature spaces. Uniform combinations of kernels can be implemented with any kernel method; VBpMKL also permits deeper analysis of the information overlap between kernels by learning a combined kernel that maximises the model likelihood. During the weight estimation process different configurations are not validated against test data and thus may not always lead to highest performance.

The fixed uniform combination of high quality kernels leads to a spreading of similarities between documents that may enable easier classification. Figure 7.8 shows a single BEAGLE-based kernel contrasted with the combined kernel. Semantic kernels have the effect of increasing the similarity between the documents, but they still keep the similarity within a relatively limited range. The single Gaussian kernel has the range of values between 0.88 and 1. The heat map of the combined kernel demonstrates the increased

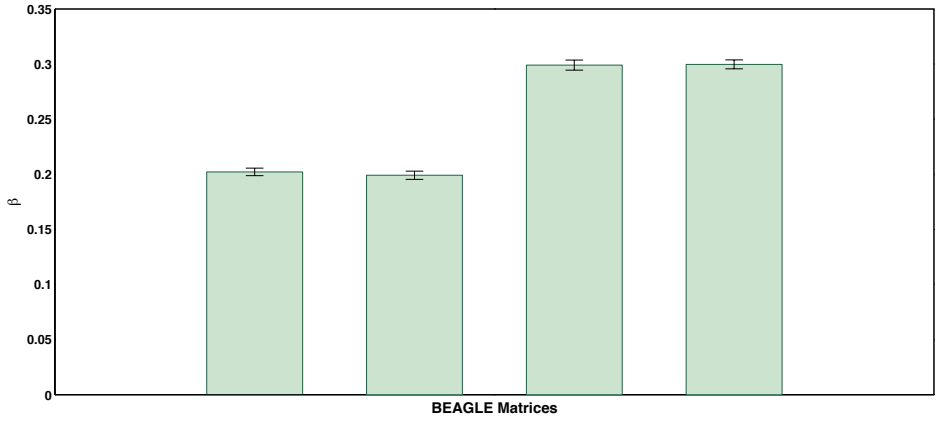


Figure 7.7: Betas for different AImed BEAGLE T1 semantic kernels. The first two kernels are created from GENIA while the last two are from OAA. The first and third kernel have $D = 2048$, while the second and fourth have $D = 4096$.

variation of the similarity values. By not normalising the resulting kernel, the range of similarities is exaggerated to span between 2.35 and 4.

The convex linear method also extends the range of similarities from $0.88 - 1$ to $0.65 - 1$; however, the normalising constraint that requires all the weights β to sum to 1, limits the spreading of the points.

Combinations of semantic kernels have lead to an increase in the performance over the initial GP classification scores presented in Chapter 5. The small, yet significant, improvement in the AUC was mirrored by a larger increase in the F-score. The increase in the AUC is an indication of a shift in ranking that resulted in a movement of positive documents ahead of some negative ones. The rise in F-score is an indication that the probabilities assigned to positive documents have also increased, leading to a higher precision and recall at the cut off probability of 0.5.

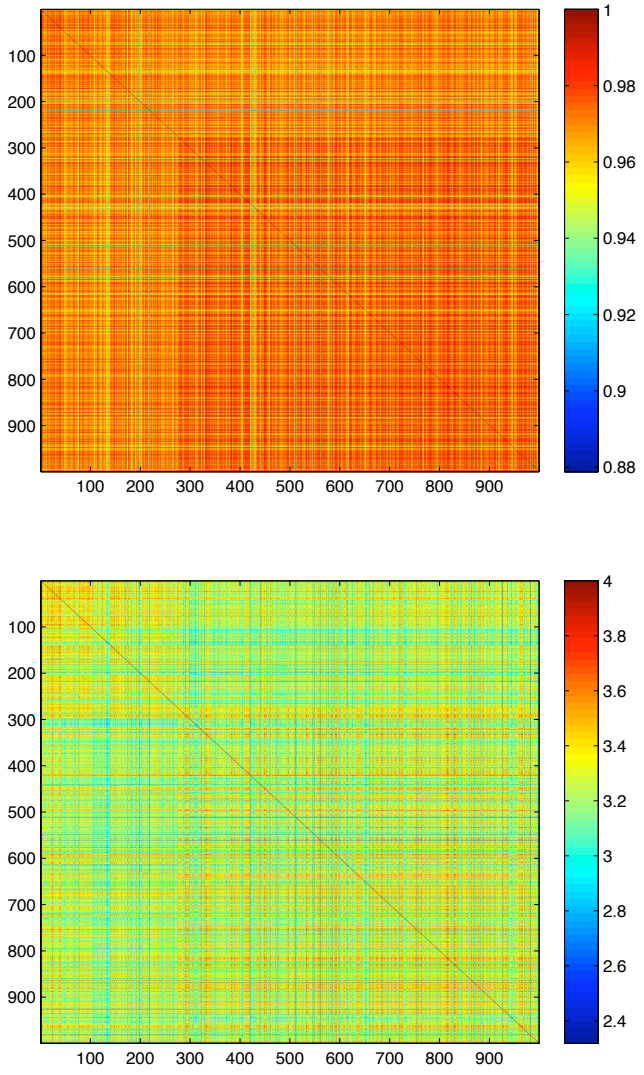


Figure 7.8: A single Gaussian BEAGLE-based kernel (top) and a combination of all four BEAGLE-based kernels (bottom) on the BC data. The x and y axes represent the documents in the collection.

BC Protein F4

Original	Single	Combination
F=0.4226 $\pm$ 0.0361	F=0.5526 $\pm$ 0.0103	**F=0.6217 $\pm$ 0.0105**
E=14.8369 $\pm$ 1.0362	E=13.4632 $\pm$ 0.3231	E=11.6227 $\pm$ 0.3225
P=0.6757 $\pm$ 0.0515	P=0.6548 $\pm$ 0.0149	P=0.7048 $\pm$ 0.0114
R=0.3235 $\pm$ 0.0373	R=0.4943 $\pm$ 0.0115	R=0.5720 $\pm$ 0.0131
A=0.9227 $\pm$ 0.0077	A=0.9224 $\pm$ 0.0027	**A=0.9339 $\pm$ 0.0024**

AImed Protein F4

Original	Single	Combination
F=0.6712 $\pm$ 0.0161	F=0.7184 $\pm$ 0.0044	**F=0.7424 $\pm$ 0.0041**
E=18.4464 $\pm$ 0.8169	E=16.8273 $\pm$ 0.2564	E=15.8507 $\pm$ 0.2528
P=0.7480 $\pm$ 0.0209	P=0.7471 $\pm$ 0.0060	P=0.7491 $\pm$ 0.0057
R=0.6128 $\pm$ 0.0198	R=0.6953 $\pm$ 0.0054	R=0.7399 $\pm$ 0.0058
A=0.9024 $\pm$ 0.0063	A=0.9052 $\pm$ 0.0021	**A=0.9113 $\pm$ 0.0021**

Table 7.4: The best results from the kernel combinations show a statistically significant improvement, over the original results, in the AUC and F-score for both of the algorithms (BC AUC $p = 7.22e^{-04}$, BC F-score $p = 8.63e^{-24}$, AImed AUC $p = 0.0010$, AImed F-score $p = 3.23e^{-20}$). The best performance for the BC data comes from a uniform combination of BEAGLE matrices, while for the AImed it comes from a uniform combination of all BEAGLE matrices and the best performing HAL matrix.

Chapter 8

Conclusion

This thesis presents a novel method for semantic smoothing of kernels when manually compiled word-similarity data is not available. This method is used to address the task of the detection of sentences in biomedical publications that describe interactions between proteins.

The task of protein-protein interaction (PPI) detection is motivated by the need for automatic methods that aid researchers and database curators in finding vital links between proteins. Proteins are a key component of cells and play part in regulating vital processes that are essential for the understanding of biological systems, diseases, and possible cures; and, while PPI detection is an important task, it is also a computationally difficult one.

A major bottleneck in the improvement of PPI detection is lack of sample training data essential for the development of accurate methods. The method proposed in this thesis addresses this lack of data by leveraging the available small datasets with large amounts of unlabelled data. Instead of doing this in the usual manner adopted by semi-supervised learning algorithms, the unlabelled data is processed by unsupervised lexical co-occurrence models and then used to smooth the training data in kernel classifiers. Each of the chapters in Part II corresponds to a step in the development of this method.

Chapter 5 demonstrated the suitability of the chosen algorithms for this task. The GP and VBpMKL classifiers were compared to the popularly used naïve Bayes and the state-

of-the-art SVM. Through a large series of experiments it was determined that GPs and VBpMKL are equivalent to the SVMs provided the right choice of features and kernel settings for each. These algorithms are probabilistic in nature and thus provide more informative output than the SVMs, but they also outperform the NB classifier.

Having shown that these algorithms are adequate for the purpose, and also having established a stringent baseline, the following chapter, introduces a way to extend the basic kernel classification with semantic information. Chapter 6 describes how information about word usage in related texts can be used to change the weights of document similarities in the classification kernel. The experiments show that this method can improve upon the best results from the previous chapter, although this change is not significant. The main contributions of this chapter are the new methodology for using unlabelled text data for semi-supervised learning, as well as a new method for separating word meanings using LDA and HAL.

In Chapter 7 the best results from the previous chapter are combined using VBpMKL to produce a statistically significant improvement upon the baseline, *i.e.* the highest scores that could be achieved with GPs using best kernel settings and features. The chapter also demonstrates several ways in which the semantic kernels, introduced for the first time in Chapter 6, could be combined, a technique that is likewise novel.

All together these contributions were used to verify the (five) hypotheses outlined in the introduction of this thesis:

1. Investigation of the semantic kernels required variations in lexical model settings. In order to focus the investigations onto the kernels and away from the classification method, an alternative to the state-of-the-art SVM classifier was studied. The SVM classifier contains an essential parameter that needs to be tuned for each new dataset and kernel, and this tuning process can increase the number of experiments by tenfold. The Gaussian process (GP) classifier is a probabilistic analogue to the SVM that does not have an equivalent parameter, and thus would make an efficient alternative, provided it has competitive performance.

In Chapter 5 the performance of GPs and SVMs was compared in an extensive series of experiments. The conclusion was that given the right choice of kernel, SVMs and GPs have statistically equivalent performance, as judged by the area under the ROC curve. Another probabilistic algorithm similar to the GPs, VBpMKL, was show to have performance equivalent to GPs on the cosine kernel.

2. The GP and VBpMKL approaches have additional benefits conveyed by the fact that they are fully probabilistic models. The Bayesian framework allows for further extensions that may be beneficial for biomedical text classification, such as the multiclass, semi-supervised, and multiexpert GP classifiers, which were discussed in detail in Chapter 3.

3. The probabilistic output of these algorithms also gives an accurate representation of model confidence in class membership, which can be used to provide users with informative rankings of new documents.

4. Chapter 6 describes the theory behind the semantic kernel method and demonstrates that it can be used to significantly improve the quality of predictions, as demonstrated by a large increase in F-score for sentence datasets. It also provides a small improvement in the ranking of documents, as demonstrated by the increase in the AUC.

   This chapter also includes a novel approach for the analysis of lexical semantic models, through induction of topics using LDA. The topics produced from a co-occurrence matrix (which was used to create a semantic kernel that lead to an increase in classification performance) show higher coherence than ones induced from a document-based LDA model.

5. The choice of VBpMKL as a second alternative to the SVM also enables the training of a single classifier from multiple kernels. In this thesis a single training dataset is smoothed using different word-similarity matrices to produce multiple kernels. These can then provide alternative views, without requiring more labelled data.

Combinations of classification-improving semantic kernels lead to statistically significant increases in both the AUC and the F-score for the sentence data.

Apart from yielding an increase in performance, the VBpMKL can also be used for automatic estimation of the contributions of individual kernels.This method provides an in depth analysis and comparison of different kernels, feature spaces, and settings

The above research has already lead to several peer-reviewed publications including: Polajnar et al. (2009b), Polajnar et al. (2009a), Polajnar and Girolami (2009a), Polajnar and Girolami (2009b), Polajnar et al. (2010), and Rogers et al. (2010). however, it is an investigation that opens up possibilities of further research in multiple directions.

8.1 Future work

The semantic method investigations carried out in this thesis can be extended in several directions: improving the semantic methods, improving the kernel classification and combination strategies, or using the models in new applications. These extensions can be tested against the large number of results presented in this thesis.

Firstly, as was noted in Chapter 4, HAL and BEAGLE could be improved using filtering techniques to discard context words with very high and very low frequencies. In essence, semantic kernels are just a combination of a document-feature matrix and a semantic model capable of collecting statistical usage information for these particular features; therefore, other semantic models could also be examined. HAL and BEAGLE are suitable representations for word-based features; however, as Padó and Lapata (2007) show, there are many different types of co-occurrence models. In particular, Lin (1998) and Padó and Lapata (2007) introduce dependency-based semantic models. Dependency tree features are often used in full PPI extraction, for example by Bunescu et al. (2005), Erkan et al. (2007), and Airola et al. (2008). An interesting avenue for future exploration would involve developing a dependency-based semantic kernel, and investigating

the effects it would have in the detection of interacting pairs.

On the other hand, if the simplest features are retained, one could follow Giuliano et al. (2006), who are able to use bag-of-words representation of the AImed dataset to detect not only the PPI sentences, but the PPI interacting pairs, themselves. Their approach uses an SVM classifier trained with fixed combinations of shallow-feature kernels, which encompass small chunks of context around the potential interacting pairs, as well as other orthographic and linguistic information. This approach can be enhanced by the research from this thesis. The contextual kernels could be enriched with semantic information, while the kernel combination can be performed with VBpMKL. This would allow for kernel combination estimation, and additional analysis of the more diverse set of kernels described in their approach.

Finally, the semantic kernel method introduced in this thesis is not limited to this particular task or domain, and thus future directions may explore the applicability of this model to different document classification tasks. The method seems to be more suited for improving the classification of shorter documents than the long ones. Short documents, as presented to the classification algorithm, are in fact just very sparse vectors, and it may be possible to generalise the semantic kernel approach to other tasks by reformulating the problem in this way.

In particular it is possible to extend this approach beyond the laboratory setting, by building applications that directly take advantage of the techniques introduced in this thesis. One obvious application is a search engine that allows biologists to identify sentences that describe PPIs. However, the semantic methods could be applied in more subtle ways. For example, the combination of LDA and HAL in Chapter 6 has shown promising results that could be further developed. The main value in this technique is its apparent ability to distinguis between different connotations of word usage. As was show in the chapter, one could distinguish between the usage of word cancer in the context of drug trials and in vitro experiments. This is an invaluable tool that could be used in other domains where word connotation is key in distinguishing relevance. A task like document moderation

for children's documents, for example, is a difficult one to do even with plenty of data. The difference between a news item that is appropriate for children and one which is not may be found, not only in the words themselves, but in the cultural connotations of these words. We can use the HAL-LDA method to build a classification tool that will assign words their different connotations and thus provide more information for classification of the documents.

These are some of the possible avenues of further research that could be undertaken on the basis of the research presented in this thesis.

Bibliography

Abney, S. (2007). *Semisupervised Learning for Computational Linguistics*. Chapman & Hall/CRC.

Abramovitch, R., Tavor, E., Jacob-Hirsch, J., Zeira, E., Amariglio, N., Pappo, O., Rechavi, G., Galun, E., and Honigman, A. (2004). A Pivotal Role of Cyclic AMP-Responsive Element Binding Protein in Tumor Progression. *Cancer Res*, 64(4):1338–1346.

Achlioptas, D. (2001). Database-friendly random projections. In *PODS '01: Proceedings of the twentieth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 274–281, New York, NY, USA. ACM.

Airola, A., Pyysalo, S., Björne, J., Pahikkala, T., Ginter, F., and Salakoski, T. (2008). All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. *BMC bioinformatics*, 9 Suppl 11.

Aizerman, A., Braverman, E. M., and Rozoner, L. I. (1964). Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837.

Albert, J. H. and Chib, S. (1993). Bayesian analysis of binary and polychotomous response data. *Journal of the American Statistical Association*, 88(422):669.

Albert, S., Gaudan, S., Knigge, H., Raetsch, A., Delgado, A., Huhse, B., Kirsch, H., Albers, M., Rebholz-Schuhmann, D., and Koegl, M. (2003). Computer-assisted generation of a protein-interaction database for nuclear receptors. *Journal of Molecular Endocrinology*, 8(17):1555–67. http://www.ebi.ac.uk/Rebholz/publications.html.

Alberts, B., Johnson, A., Lewis, J., Raff, M., Roberts, K., and Walter, P. (2002). *Molecular Biology of the Cell, Fourth Edition*. Garland.

Alex, B., Grover, C., Haddow, B., Kabadjov, M., Klein, E., Matthews, M., Roebuck, S., Tobin, R., and Wang, X. (2008a). Assisted curation: Does text mining really help? In *Proceedings of Pacific Symposium on Biocomputing*, Hawaii, USA.

Alex, B., Grover, C., Haddow, B., Kabadjov, M., Klein, E., Matthews, M., Roebuck, S., Tobin, R., and Wang, X. (2008b). The ITI TXM corpora: Tissue expressions and protein-protein interactions. In *Proceedings of LREC*, volume 8.

Alex, B., Grover, C., Haddow, B., Kabadjov, M., Klein, E., Matthews, M., Tobin, R., and Wang, X. (2008c). Automating curation using a natural language processing pipeline. *Genome Biology*, 9(Suppl 2).

Alex, B., Haddow, B., and Grover, C. (2007). Recognising nested named entities in biomedical text. In *Biological, translational, and clinical language processing*, pages 65–72, Prague, Czech Republic. Association for Computational Linguistics.

Alfarano, C., Andrade, C. E., Anthony, K., Bahroos, N., Bajec, M., Bantoft, K., Betel, D., Bobechko, B., Boutilier, K., Burgess, E., Buzadzija, K., Cavero, R., D'Abreo, C., Donaldson, I., Dorairajoo, D., Dumontier, M. J., Dumontier, M. R., Earles, V., Farrall, R., Feldman, H., Garderman, E., Gong, Y., Gonzaga, R., Grytsan, V., Gryz, E., Gu, V., Haldorsen, E., Halupa, A., Haw, R., Hrvojic, A., Hurrell, L., Isserlin, R., Jack, F., Juma, F., Khan, A., Kon, T., Konopinsky, S., Le, V., Lee, E., Ling, S., Magidin, M., Moniakis, J., Montojo, J., Moore, S., Muskat, B., Ng, I., Paraiso, J. P., Parker, B., Pintilie, G., Pirone, R., Salama, J. J., Sgro, S., Shan, T., Shu, Y., Siew, J., Skinner, D., Snyder, K., Stasiuk, R., Strumpf, D., Tuekam, B., Tao, S., Wang, Z., White, M., Willis, R., Wolting, C., Wong, S., Wrong, A., Xin, C., Yao, R., Yates, B., Zhang, S., Zheng, K., Pawson, T., Ouellette, B. F., and Hogue, C. W. (2005). The biomolecular interaction network database and related tools 2005 update. *Nucleic Acids Res*, 33(Database issue):418–424.

Altun, Y., Hofmann, T., and Smola, A. J. (2004). Gaussian process classification for segmenting and annotating sequences. In *Proceedings of the Twenty-First international Conference on Machine Learning, ICML '04*, volume 69. ACM, New York, NY.

Ashburner, M., Ball, C. A., Blake, J. A., Botstein, D., Butler, H., Cherry, J. M., Davis, A. P., Dolinski, K., Dwight, S. S., Eppig, J. T., Harris, M. A., Hill, D. P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J. C., Richardson, J. E., Ringwald, M., Rubin, G. M., and Sherlock, G. (2000). Gene ontology: tool for the unification of biology. the gene ontology consortium. *Nat Genet*, 25(1):25–29.

Aslam, J. A., Yilmaz, E., and Pavlu, V. (2005). A geometric interpretation of r-precision and its correlation with average precision. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 573–574, New York, NY, USA. ACM.

Azzopardi, L., Girolami, M., and Crowe, M. (2005). Probabilistic hyperspace analogue to language. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 575–576, New York, NY, USA. ACM.

Baldwin, B. and Carpenter, B. (2008). Lingpipe software package. WWW document.

Basili, R., Cammisa, M., and Moschitti, A. (2005). A semantic kernel to exploit linguistic knowledge. In *AI*IA 2005: Advances in Artificial Intelligence*, pages 290–302.

Bennett, P. (2000). Assessing the calibration of naive bayes' posterior estimates. Technical report, School of Computer Science, Carnegie Mellen University.

Bingham, E. and Mannila, H. (2001). Random projection in dimensionality reduction: applications to image and text data. In *KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 245–250, New York, NY, USA. ACM.

Bishop, C. M. (2006). *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer.

Blaschke, C. and Valencia, A. (2001). The potential use of SUISEKI as a protein interaction discovery tool. *Genome Informatics*, 12:123–134.

Blei, D. M., Franks, K., Jordan, M. I., and Mian, I. S. (2006). Statistical modeling of biomedical corpora: mining the caenorhabditis genetic center bibliography for genes related to life span. *BMC Bioinformatics*, 7:250–250.

Blei, D. M., Ng, A. Y., and Jordan, M. I. (2003). Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022.

Boser, B. E., Guyon, I., and Vapnik, V. (1992). A training algorithm for optimal margin classifiers. In *Computational Learing Theory*, pages 144–152.

Buckley, C. and Voorhees, E. M. (2000). Evaluating evaluation measure stability. In *SIGIR '00: Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 33–40, New York, NY, USA. ACM.

Bunescu, R., Ge, R., Kate, R. J., Marcotte, E. M., Mooney, R. J., Ramani, A. K., and Wong, Y. W. (2005). Comparative experiments on learning information extractors for proteins and their interactions. *Artif Intell Med*, 33(2):139–155.

Burgess, C. and Conley, P. (1998). Developing semantic representations for proper names. In *Proceedings of the Cognitive Science Society*, pages p.185–190.

Burgess, C., Livesay, K., and Lund, K. (1998). Explorations in context space: Words, sentences, discourse. *Discourse Processes*, 25:211 – 257.

Burgess, C. and Lund, K. (1997). Modeling parsing constraints with high-dimensional context space. *Language and Cognitive Processes*, 12:177–210.

Chai, K. M. A., Chieu, H. L., and Ng, H. T. (2002). Bayesian online classifiers for text classification and filtering. In *SIGIR '02: Proceedings of the 25th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 97–104, New York, NY, USA. ACM Press.

Chang, L. and Karin, M. (2001). Mammalian map kinase signalling cascades. *Nature*, 410(6824):37–40.

Chapelle, O., Schölkopf, B., and Zien, A., editors (2006). *Semi-Supervised Learning*. MIT Press, Cambridge, MA.

Chen, H. and Sharp, B. M. (2004). Content-rich biological network constructed by mining pubmed abstracts. *BMC Bioinformatics*, 5:147–147.

Chu, W. and Ghahramani, Z. (2005a). Gaussian processes for ordinal regression. *Journal of Machine Learning Research*, 6:1019–1041.

Chu, W. and Ghahramani, Z. (2005b). Preference learning with gaussian processes. In *Proceedings of the 22nd international Conference on Machine Learning (Bonn, Germany, August 07 - 11, 2005). ICML '05, vol. 119*, pages 137–144.

Chu, W., Ghahramani, Z., Falciani, F., and Wild, D. L. (2005). Biomarker discovery in microarray gene expression data with gaussian processes. *Bioinformatics*, 21(16):3385–3393.

Clegg, A. B. and Shepherd, A. J. (2007). Benchmarking natural-language parsers for biological applications using dependency graphs. *BMC Bioinformatics*, 8:24–24.

Cockerill, M. (2008). Data mining open access research. *Open Access Now*.

Cohen, A. M. and Hersh, W. R. (2005). A survey of current work in biomedical text mining. *Briefings in Bioinformatics*, 6(1):51–71.

Cohen, K. B., Fox, L., Ogren, P. V., and Hunter, L. (2005). Corpus design for biomedical natural language processing. In *Proceedings of the ACL-ISMB workshop on linking*

biological literature, ontologies and databases: mining biological semantics, pages 38–45.

Cole, R., editor (1997). *Survey of the state of the art in human language technology*. Cambridge University Press, New York, NY, USA.

Collier, N., Park, H., Ogata, N., Tateisi, Y., Nobata, C., Ohta, T., Sekimizu, T., Imai, H., Ibushi, K., and Tsujii, J. (1999). The GENIA project: Corpus-based knowledge acquisition and information extraction from genome research papers.

Cortes, C. and Mohri, M. (2004). Confidence intervals for the area under the roc curve. In *NIPS*.

Crammer, K. and Singer, Y. (2001). On the algorithmic implementation of multiclass kernel-based vector machines. *Journal of Machine Learning Research*, 2:265–292.

Craven, M. and Kumlien, J. (1999). Constructing biological knowledge bases by extracting information from text sources. In *Proceedings of the Seventh International Conference on Intelligent Systems for Molecular Biology*, pages 77–86, Heidelberg, Germany. AAAI Press.

Cristianini, N. and Shawe-Taylor, J. (2000). *An Introduction to Support Vector Machines and other kernel-based learning methods*. Cambridge University Press, Cambridge.

Cristianini, N., Shawe-Taylor, J., and Lodhi, H. (2002). Latent semantic kernels. *J. Intelligent Information Systems*, 18(2-3):127–152.

Cussens, J. and Nédellec, C., editors (2005). *Proceedings of the 4th Learning Language in Logic Workshop (LLL05)*, Bonn.

Dagan, I., Lee, L., and Pereira, F. C. N. (1999). Similarity-based models of word cooccurrence probabilities. *Mach. Learn.*, 34(1-3):43–69.

Damoulas, T. and Girolami, M. A. (2008). Probabilistic multi-class multi-kernel learning: On protein fold recognition and remote homology detection. *Bioinformatics*.

Davis, J. and Goadrich, M. (2006). The relationship between precision-recall and roc curves. In *ICML '06: Proceedings of the 23rd international conference on Machine learning*, pages 233–240, New York, NY, USA. ACM.

Ding, C. H. and Dubchak, I. (2001). Multi-class protein fold recognition using support vector machines and neural networks. *Bioinformatics*, 17(4):349–358.

Ding, J., Berleant, D., Nettleton, D., and Wurtele, E. (2002). Mining MEDLINE: Abstracts, sentences, or phrases? http://helix-web.stanford.edu/psb02/ding.pdf.

Donaldson, I., Martin, J., de Bruijn, B., Wolting, C., Lay, V., Tuekam, B., Zhang, S., Baskin, B., Bader, G. D., Michalickova, K., Pawson, T., and Hogue, C. W. (2003). PreBIND and Textomy - mining the biomedical literature for protein-protein interactions using a support vector machine. *BMC Bioinformatics*, 4(11).

Eikmeyer, H.-J. and Rieser, H. (1981). The notional category of modality. In Eikmeyer, H.-J. and Rieser, H., editors, *Words, Worlds, and Contexts : New Approaches in Word Semantics*, pages 1–20. W. de Gruyter.

Erkan, G., Ozgur, A., and Radev, D. R. (2007). Semi-supervised classification for extracting protein interaction sentences using dependency parsing. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*, pages 228–237.

Fellbaum, C. et al. (1998). *WordNet: An electronic lexical database*. MIT press, Cambridge, MA.

Fradkin, D. and Madigan, D. (2003). Experiments with random projections for machine learning. In *KDD '03: Proceedings of the ninth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 517–522, New York, NY, USA. ACM.

Friedman, C., Kra, P., Yu, H., Krauthammer, M., and Rzhetsky, A. (2001). Genies: a natural-language processing system for the extraction of molecular pathways from journal articles. *Bioinformatics*, 17 Suppl. 1:S74–S82.

Fu, W., Sanders-Beer, B. E., Katz, K. S., Maglott, D. R., Pruitt, K. D., and Ptak, R. G. (2009). Human immunodeficiency virus type 1, human protein interaction database at ncbi. *Nucleic Acids Res*, 37(Database issue):417–422.

Gärdenfors, P. (2004). Conceptual spaces as a framework for knowledge representation. *Mind and Matter*, 2:9–27.

Girolami, M. and Rogers, S. (2006). Variational bayesian multinomial probit regression with gaussian process priors. *Neural Computation*, 18(8):1790–1817.

Girolami, M. and Zhong, M. (2007). Data integration for classification problems employing gaussian process priors. In Schölkopf, B., Platt, J., and Hoffman, T., editors, *Advances in Neural Information Processing Systems 19*, pages 465–472. MIT Press, Cambridge, MA.

Giuliano, C., Lavelli, A., and Romano, L. (2006). Exploiting shallow linguistic information for relation extraction from biomedical literature. In *Proceedings of EACL 2006*, pages 401–408.

Gorban, A., Kegl, B., Wunsch, D., and Zinovyev, A., editors (2007). *Principal Manifolds for Data Visualization and Dimension Reduction (Lecture Notes in Computational Science and Engineering)*. Springer, 1 edition.

Hakenberg, J., Leser, U., Kirsch, H., and Rebholz-Schuhmann, D. (2006). Collecting a large corpus from all of MEDLINE. In *Proc. of Second International Symposium on Semantic Mining in Biomedicine (SMBM)*, Jena, Germany.

Hanisch, D., Fluck, J., Mevissen, H. T., and Zimmer, R. (2003). Playing biology's name game: identifying protein names in scientific text. *Pac Symp Biocomput*, pages 403–414.

Hao, Y., Zhu, X., Huang, M., and Li, M. (2005). Discovering patterns to extract protein-protein interactions from the literature: Part II. *Bioinformatics*, 21(15):3294–3300.

He, F. and Ding, X. (2007). Improving naive bayes text classifier using smoothing methods. *Advances in Information Retrieval*, pages 703–707.

Hersh, W. (2005). Evaluation of biomedical text mining systems: lessons learned from information retrieval. *Briefings in Bioinformatics*, 6:344–356.

Hersh, W. and Hickam, D. H. (1998). How well do physicians use electronic information retrieval systems? a framework for investigation and systematic review. *Journal of the American Medical Association*, 280:1347–1352.

Hersh, W. and Voorhees, E. (2009). Trec genomics special issue overview. *Inf. Retr.*, 12(1):1–15.

Hersh, W. R. (2008). *Information Retrieval: A Health and Biomedical Perspective*. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 3rd edition.

Hirschman, L., Colosimo, M., Morgan, A., and Yeh, A. (2005a). Overview of biocreative task 1b: normalized gene lists. *BMC Bioinformatics*, 6 Suppl 1.

Hirschman, L., Morgan, A. A., and Yeh, A. S. (2002). Rutabaga by any other name: extracting biological names. *J Biomed Inform*, 35(4):247–259.

Hirschman, L., Yeh, A., Blaschke, C., and Valencia, A. (2005b). Overview of BioCreAtIvE: critical assessment of information extraction for biology. *BMC Bioinformatics*, 6 Suppl 1.

Hoffmann, R. and Valencia, A. (2004). A gene network for navigating the literature. *Nat Genet*, 36(7):664–664.

Hofmann, T. (1999). Probabilistic latent semantic indexing. In *SIGIR '99: Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, pages 50–57, New York, NY, USA. ACM.

Hsieh, H. C., Hsieh, Y. H., Huang, Y. H., Shen, F. C., Tsai, H. N., Tsai, J. H., Lai, Y. T., Wang, Y. T., Chuang, W. J., and Huang, W. (2005). Hhr23a, a human homolog of saccharomyces cerevisiae rad23, regulates xeroderma pigmentosum c protein and is required for nucleotide excision repair. *Biochem Biophys Res Commun*, 335(1):181–187.

Hsu, C.-W. and Lin, C.-J. (2002). A comparison of methods for multiclass support vector machines. *IEEE Transactions on Neural Networks*, 13:415–425.

Jenssen, T. K., Laegreid, A., Komorowski, J., and Hovig, E. (2001). A literature network of human genes for high-throughput analysis of gene expression. *Nat Genet*, 28(1):21–28.

Jin, X. and Bie, R. (2006). Improving software quality classification with random projection. In *Proceedings of ICCI 2006: 5th IEEE International Conference on Cognitive Informatics*, volume 1, pages 149–154.

Joachims, T. (1998). Text categorization with support vector machines: Learning with many relevant features. In *Proceedings of the European Conference on Machine Learning*, pages 137–142. Springer, Chemnitz, Germany.

Joachims, T. (1999). *Advances in Kernel Methods - Support Vector Learning*, chapter Making large-Scale SVM Learning Practical. MIT-Press, Cambridge, Massachusetts.

Jones, M. N., Kintsch, W., and Mewhort, D. J. (2006). High-dimensional semantic space accounts of priming. *Journal of Memory and Language*, 55(4):534–552.

Jones, M. N. and Mewhort, D. J. K. (2007). Representing word meaning and order information in a composite holographic lexicon. *Psychological Review*, 114:1–37.

Kakkonen, T., Myller, N., Timonen, J., and Sutinen, E. (2005). Automatic essay grading with probabilistic latent semantic analysis. In *EdAppsNLP 05: Proceedings of the second workshop on Building Educational Applications Using NLP*, pages 29–36, Morristown, NJ, USA. Association for Computational Linguistics.

Kanehisa, M., Goto, S., Furumichi, M., Tanabe, M., and Hirakawa, M. (2010). Kegg for representation and analysis of molecular networks involving diseases and drugs. *Nucleic Acids Res*, 38(Database issue):355–360.

Kaski, S. (1998). Dimensionality reduction by random mapping: fast similarity computation for clustering. In *Proceedings of the 1998 IEEE International Joint Conference on Neural Networks*, volume 1, pages 413–418 vol.1.

Katrenko, S. and Adriaans, P. (2006). Learning relations from biomedical corpora using dependency trees. *Knowledge Discovery and Emergent Complexity in BioInformatics, Lecture Notes in Computer Science*, 4366.

Keerthi, S. S., Chapelle, O., and DeCoste, D. (2006). Building support vector machines with reduced classifier complexity. *Journal of Machine Learning Research*, 7:14931515.

Kerrien, S., Alam-Faruque, Y., Aranda, B., Bancarz, I., Bridge, A., Derow, C., Dimmer, E., Feuermann, M., Friedrichsen, A., Huntley, R., Kohler, C., Khadake, J., Leroy, C., Liban, A., Lieftink, C., Montecchi-Palazzi, L., Orchard, S., Risse, J., Robbe, K., Roechert, B., Thorneycroft, D., Zhang, Y., Apweiler, R., and Hermjakob, H. (2007). Intact–open source resource for molecular interaction data. *Nucleic Acids Res*, 35(Database issue):561–565.

Kim, J. D., Ohta, T., Tateisi, Y., and Tsujii, J. (2003). GENIA corpus–semantically annotated corpus for bio-textmining. *Bioinformatics*, 19 Suppl 1:180–182.

Kim, S., Yoon, J., and Yang, J. (2008). Kernel approaches for genic interaction extraction. *Bioinformatics*, 24(1):118–126.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *IJCAI*, pages 1137–1145.

Koster, C., Seibert, O., and Seutter, M. (2006). The phasar search engine. In *Proceedings NLDB 2006*, pages 141–152. Springer LNCS 3999.

Koster, C., Seutter, M., and O.Seibert (2007). Parsing the medline corpus. In *Proceedings RANLP 2007*, pages 325–329.

Koster, C. H., Oostdijk, N., Verberne, S., and d'Hondt, E. (2009). Challenges in Professional Search with PHASAR. In *Proceedings of Dutch-Belgium Information Retrieval workshop (DIR) 2009*, pages 101–102.

Kotsiantis, S. B. (2007). Supervised machine learning: A review of classification techniques. *Informatica*, 31(3):249–268.

Krallinger, M., Erhardt, R. A., and Valencia, A. (2005). Text-mining approaches in molecular biology and biomedicine. *Drug Discov Today*, 10(6):439–445.

Krallinger, M., Leitner, F., Rodriguez-Penagos, C., and Valencia, A. (2008a). Overview of the protein-protein interaction annotation extraction task of BioCreative II. *Genome Biol*, 9 Suppl 2.

Krallinger, M., Morgan, A., Smith, L., Leitner, F., Tanabe, L., Wilbur, J., Hirschman, L., and Valencia, A. (2008b). Evaluation of text-mining systems for biology: overview of the Second BioCreative community challenge. *Genome Biol*, 9 Suppl 2.

Krauthammer, M., Rzhetsky, A., Morozov, P., and Friedman, C. (2000). Using BLAST for identifying gene and protein names in journal articles. *Gene*, pages 245–252.

Lama, N. and Girolami, M. (2008). Vbmp: variational Bayesian Multinomial Probit Regression for multi-class classification in R. *Bioinformatics*, 24(1):135–136.

Landauer, T. K., Foltz, P. W., and Laham, D. (1998). An introduction to latent semantic analysis. *Discourse Processes*, 25:259–284.

Lawrence, N. and Jordan, M. (2006). Gaussian processes and the null-category noise model. In Chapelle, O., Schölkopf, B., and Zien, A., editors, *Semi-supervised Learning*, chapter 8. MIT Press.

Lawrence, N., Platt, J. C., and Jordan, M. I. (2005). Extensions of the informative vector machine. In Winkler, J., Lawrence, N. D., and Niranjan, M., editors, *Proceedings of the Sheffield Machine Learning Workshop*, pages 56–87, Berlin. Springer-Verlag.

Lawrence, N. D., Seeger, M., and Herbrich, R. (2003). *Advances in Neural Information Processing Systems*, chapter Fast sparse Gaussian process methods: the informative vector machine, pages 625–632. MIT Press, Cambridge, Massachusets.

Lee, Y., Lin, Y., and Wahba, G. (2004). Multicategory support vector machines: Theory and application to the classification of microarray data and satellite radiance data. *Journal of the American Statistical Association*, 99:67–81(15).

Lewis, D. D. (1995). Evaluating and Optimizing Autonomous Text Classification Systems. In Fox, E. A., Ingwersen, P., and Fidel, R., editors, *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 246–254, Seattle, Washington. ACM Press.

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. In *ECML '98: Proceedings of the 10th European Conference on Machine Learning*, pages 4–15, London, UK. Springer-Verlag.

Lin, D. (1998). Dependency-based evaluation of MINIPAR. In *Workshop on the Evaluation of Parsing Systems*.

Lowe, W. (2001). Towards a theory of semantic space. In Moore, J. D. and Stenning, K., editors, *Proceedings of the Twenty-Third Annual Conference of the Cognitive Science Society*, pages 576–581, Mahwah NJ. Lawrence Erlbaum Associates.

Lund, K. and Burgess, C. (1996). Producing high-dimensional semantic spaces from lexical co-occurrence. *Behavior Research Methods, Instrumentation, and Computers*, 28(2):203–208.

Manning, C. D., Raghavan, P., and Schütze, H. (2008). *Introduction to Information Retrieval*. Cambridge University Press.

Manning, C. D. and Schütze, H. (1999). *Foundations of Statistical Natural Language Processing*. The MIT Press, Cambridge, Massachusetts.

Marcotte, E. M., Xenarios, I., and Eisenberg, D. (2001). Mining literature for protein-protein interactions. *Bioinformatics*, 17:359 – 363.

McMurray, B. (2007). Moo-cow! Mummy! More! How do children learn so many words? *Significance*, 4(4):159–163.

Mewes, H. W., Amid, C., Arnold, R., Frishman, D., Güldener, U., Mannhaupt, G., Münsterkötter, M., Pagel, P., Strack, N., Stümpflen, V., Warfsmann, J., and Ruepp, A. (2004). Mips: analysis and annotation of proteins from whole genomes. *Nucleic Acids Res*, 32(Database issue):41–44.

Minier, Z., Bodo, Z., and Csato, L. (2007). Wikipedia-based kernels for text categorization. In *SYNASC '07: Proceedings of the Ninth International Symposium on Symbolic and Numeric Algorithms for Scientific Computing*, pages 157–164, Washington, DC, USA. IEEE Computer Society.

Miyao, Y., Ohta, T., Masuda, K., Tsuruoka, Y., Yoshida, K., Ninomiya, T., and Tsujii, J. (2006). Semantic retrieval for the accurate identification of relational concepts in massive textbases. In *ACL-44: Proceedings of the 21st International Conference on Computational Linguistics and the 44th annual meeting of the Association for Computational Linguistics*, pages 1017–1024, Morristown, NJ, USA. Association for Computational Linguistics.

Miyao, Y. and Tsujii, J. (2005). Probabilistic disambiguation models for wide-coverage hpsg parsing. In *ACL '05: Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 83–90, Morristown, NJ, USA. Association for Computational Linguistics.

Nanas, N., Roeck, A. N. D., and Vavalis, M. (2009). What happened to content-based information filtering?. In Azzopardi, L., Kazai, G., Robertson, S. E., Rger, S. M., Shokouhi, M., Song, D., and Yilmaz, E., editors, *ICTIR 2009*, volume 5766 of *Lecture Notes in Computer Science*, pages 249–256. Springer.

Nigam, K., McCallum, A., and Mitchell, T. (2006). Semi-supervised text classification using em. In Chapelle, O., Zien, A., and Scholkopf, B., editors, *Semi-Supervised Learning*. MIT Press, Boston.

Oda, K., Kim, J. D., Ohta, T., Okanohara, D., Matsuzaki, T., Tateisi, Y., and Tsujii, J. (2008). New challenges for text mining: mapping between text and manually curated pathways. *BMC Bioinformatics*, 9 Suppl 3.

Okada, H., Zhang, X., Ben Fofana, I., Nagai, M., Suzuki, H., Ohashi, T., and Shida, H. (2009). Synergistic effect of human cyct1 and crm1 on hiv-1 propagation in rat t cells and macrophages. *Retrovirology*, 6(1):43.

Osgood, C. E., Suci, G. J., and Tannenbaum, P. H. (1957). *The measurment of meaning*. Urbana: University of Illinois Press.

Padó, S. and Lapata, M. (2007). Dependency-based construction of semantic space models. *Comput. Linguist.*, 33(2):161–199.

Papadimitriou, C. H., Raghavan, P., Tamaki, H., and Vempala, S. (2000). Latent semantic indexing: a probabilistic analysis. *Journal of Computer and System Sciences*, 61(2):217–235.

Pearson, H. (2001). Biology's name game. *Nature*, 411(6838):631–632.

Platt, J. (1999). Probabilities for sv machines. In Smola, A., Bartlett, P., Schlkopf, B., and Schuurmans, D., editors, *Advances in Large Margin Classifiers*, pages 61–74. MIT Press, Cambridge, Massachusetts.

Polajnar, T., Damoulas, T., and Girolami, M. (2010). Protein interaction sentence detection using multiple semantic kernels. Under review for the International Journal of Systems Science special issue on Integrative Genomics.

Polajnar, T. and Girolami, M. (2009a). Application of lexical topic models to protein interaction sentence prediction. In *NIPS 2009 Workshop on Applications for Topic Models: Text and Beyond*, Whistler, Canada.

Polajnar, T. and Girolami, M. (2009b). Semi-supervised prediction of protein interaction sentences exploiting semantically encoded metrics. In *Proceedings of the 4th*

IAPR International Conference, Pattern Recognition in Bioinformatics, pages 270–281. Springer Verlag.

Polajnar, T., Rogers, S., and Girolami, M. (2009a). Classication of protein interaction sentences via Gaussian processes. In *Proceedings of 4th IAPR International Conference, Pattern Recognition in Bioinformatics*, pages 282–292. Springer Verlag.

Polajnar, T., Rogers, S., and Girolami, M. (2009b). Protein interaction detection in sentences via Gaussian processes: A preliminary evaluation. *International Journal of Data Mining and Bioinformatics*. To appear.

Porter, M. F. (1997). An algorithm for suffix stripping. *Readings in information retrieval*, pages 313–316.

Pyysalo, S., Airola, A., Heimonen, J., Bjorne, J., Ginter, F., and Salakoski, T. (2008). Comparative analysis of five protein-protein interaction corpora. *BMC Bioinformatics*, 9(Suppl 3):S6.

Pyysalo, S., Ginter, F., Heimonen, J., Björne, J., Boberg, J., Järvinen, J., and Salakoski, T. (2007). Bioinfer: a corpus for information extraction in the biomedical domain. *BMC Bioinformatics*, 8:50–50.

Rasmussen, C. E. and Williams, C. K. I. (2006). *Gaussian Processes for Machine Learning*. MIT Pres, Cambridge, Massachusetts.

Rebholz-Schuhmann, D., Kirsch, H., Arregui, M., Gaudan, S., Riethoven, M., and Stoehr, P. (2007). Ebimed–text crunching to gather facts for proteins from medline. *Bioinformatics*, 23(2):237–244.

Rennie, J. D. M., Shih, L., Teevan, J., and Karger, D. R. (2003). Tackling the poor assumptions of naive bayes text classifiers. In *ICML '03*.

Rifkin, R. and Klautau, A. (2004). In defense of one-vs-all classification. *J. Mach. Learn. Res.*, 5:101–141.

Rinaldi, F., Schneider, G., Kaljurand, K., Hess, M., Andronis, C., Konstandi, O., and Persidis, A. (2007). Mining of relations between proteins over biomedical scientific literature using a deep-linguistic approach. *Artif Intell Med*, 39(2):127–136.

Riordan, B. and Jones, M. N. (2007). Comparing semantic space models using child-directed speech. In MacNamara, D. S. and Trafton, J. G., editors, *Proceedings of the 29th Annual Cognitive Science Society*, pages 599–604.

Rish, I. (2001). An empirical study of the naive bayes classifier. In *IJCAI-01 workshop on "Empirical Methods in AI"*.

Roberts, P. M. (2006). Mining literature for systems biology. *Brief Bioinform*, 7(4):399–406.

Rogers, S. and Girolami, M. (2007). Multi-class semi-supervised learning with the ϵ-truncated multinomial probit gaussian process. *Journal of Machine Learning Research Workshop and Conference Proceedings*, 1:17–32.

Rogers, S., Girolami, M., and Polajnar, T. (2010). Semi-parametric analysis of multi-rater data. *Statistics and Computing*, 20:317–334. 10.1007/s11222-009-9125-z.

Rohde, D., Gonnerman, L., and Plaut, D. (2005). An improved model of semantic similarity based on lexical co-occurrence. unpublished manuscript.

Rosario, B. and Hearst, M. (2005). Multi-way relation classification: Application to protein-protein interaction. In *Proceedings of HLT-NAACL'05*.

Rzhetsky, A., Iossifov, I., Koike, T., Krauthammer, M., Kra, P., Morris, M., Yu, H., Dubou, P. A., Weng, W., and et al. (2004). GeneWays: a system for extracting, analyzing, visualizing, and integrating molecular pathway data. *Journal of Biomedical Informatics*, 37(1):43–53. http://citeseer.ist.psu.edu/rzhetsky04geneways.html.

Salwinski, L., Miller, C. S., Smith, A. J., Pettit, F. K., Bowie, J. U., and Eisenberg, D. (2004). The database of interacting proteins: 2004 update. *Nucleic Acids Res*, 32(Database issue):449–451.

Sanderson, M. and Zobel, J. (2005). Information retrieval system evaluation: effort, sensitivity, and reliability. In *SIGIR '05: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 162–169, New York, NY, USA. ACM.

Seeger, M. and Jordan, M. I. (2004). Sparse gaussian process classification with multiple classes. Technical Report TR 661, Department of Statistics, University of California at Berkeley.

Sekimizu, T., Park, H., and Tsujii, J. (1998). Identifying the interaction between genes and gene products based on frequently seen verbs MEDLINE abstracts.

Shawe-Taylor, J. and Cristianini, N. (2004). *Kernel Methods for Pattern Analysis*. Cambridge University Press, New York, NY, USA.

Sherry, B., Tekamp-Olson, P., Gallegos, C., Bauer, D., Davatelis, G., Wolpe, S. D., Masiarz, F., Coit, D., and Cerami, A. (1988). Resolution of the two components of macrophage inflammatory protein 1, and cloning and characterization of one of those components, macrophage inflammatory protein 1 beta. *J Exp Med*, 168(6):2251–2259.

Silva, Catarina, Ribeiro, and Bernardete (2007). On text-based mining with active learning and background knowledge using SVM. *Soft Computing*, 11(6):519–530.

Smith, L., Tanabe, L. K., Ando, R. J., Kuo, C. J., Chung, I. F., Hsu, C. N., Lin, Y. S., Klinger, R., Friedrich, C. M., Ganchev, K., Torii, M., Liu, H., Haddow, B., Struble, C. A., Povinelli, R. J., Vlachos, A., Baumgartner, W. A., Hunter, L., Carpenter, B., Tsai, R. T., Dai, H. J., Liu, F., Chen, Y., Sun, C., Katrenko, S., Adriaans, P., Blaschke, C., Torres, R., Neves, M., Nakov, P., Divoli, A., Maña-López, M., Mata, J., and Wilbur, W. J. (2008). Overview of BioCreative II gene mention recognition. *Genome Biol*, 9 Suppl 2.

Song, D. and Bruza, P. D. (2001). Discovering information flow using a high dimensional conceptual space. In *Proceedings of ACM SIGIR 2001*, pages 327–333.

Song, Y., Zhang, L., and Giles, C. L. (2008). A sparse gaussian processes classification framework for fast tag suggestions. In *Proceedings of the 17th Conference on Information and Knowledge Management*.

Stankovic, M., Moustakis, V., and Stankovic, S. (2005). Text categorization using informative vector machine. In *The International Conference on Computer as a Tool, 2005. EUROCON 2005*, pages 209 – 212.

Subramaniam, L. V., Mukherjea, S., Kankar, P., Srivastava, B., Batra, V. S., Kamesam, P. V., and Kothari, R. (2003). Information extraction from biomedical literature: methodology, evaluation and an application. In *CIKM '03: Proceedings of the twelfth international conference on Information and knowledge management*, pages 410–417, New York, NY, USA. ACM Press.

Sugiyama, K., Hatano, K., and Masatoshi Yoshikawa, S. U. (2003). Extracting information on protein-protein interactions from biological literature based on machine learning approaches. In Gribskov, M., Kanehis, M., Miyano, S., and Takagi, T., editors, *Genome Informatics 2003*, pages 701–702. Universal Academy Press, Tokyo.

Szedmak, S., Shawe-Taylor, J., and Parado-Hernandez, E. (2006). Learning via linear operators: Maximum margin regression. Technical report, University of Southampton, UK.

Tanabe, L. and Wilbur, W. J. (2002). Tagging gene and protein names in full text articles. In *Proceedings of the Workshop on Natural Language Processing in the Biomedical Domain*, pages 9–13, Philadelphia. Association for Computational Linguistics. http://bioinformatics.oxfordjournals.org/cgi/content/abstract/18/8/1124.

Tanabe, L., Xie, N., Thom, L. H., Matten, W., and Wilbur, W. J. (2005). GENETAG: a tagged corpus for gene/protein named entity recognition. *BMC Bioinformatics*, 6 Suppl 1.

Thomas, J., Milward, D., Ouzounis, C., Pulman, S., and Carroll, M. (2000). Automatic extraction of protein interactions from scientific abstracts. In *Pacific Symposium on Biocomputing 5*, pages 538–549.

Van Rijsbergen, C. J. (1979). *Information Retrieval, 2nd edition*. Dept. of Computer Science, University of Glasgow.

Vapnik, V. (1995). *The Nature of Statistical Learning Theory*. Springer Verlag, New York.

Versley, Y. (2006). Disagreement dissected: Vagueness as a source of ambiguity in nominal (co-)reference. In *Proceedings of the ESSLLI 2006 Workshop on Ambiguity in Anaphora*, Malaga, Spain.

Wang, T., Li, Y., Bontcheva, K., Cunningham, H., and Wang, J. (2006). Automatic extraction of hierarchical relations from text. In *Proceedings of the Third European Semantic Web Conference (ESWC 2006)*. Springer.

Wilbur, W. J., Rzhetsky, A., and Shatkay, H. (2006). New directions in biomedical text annotation: definitions, guidelines and corpus construction. *BMC Bioinformatics*, 7:356–356.

Wu, C. H., Apweiler, R., Bairoch, A., Natale, D. A., Barker, W. C., Boeckmann, B., Ferro, S., Gasteiger, E., Huang, H., Lopez, R., Magrane, M., Martin, M. J., Mazumder, R., O'Donovan, C., Redaschi, N., and Suzek, B. (2006). The universal protein resource (UniProt): an expanding universe of protein information. *Nucleic Acids Res*, 34(Database issue):187–191.

Yang, Y. (2001). A study on thresholding strategies for text categorization. In Croft, W. B., Harper, D. J., Kraft, D. H., and Zobel, J., editors, *Proceedings of SIGIR-01, 24th ACM International Conference on Research and Development in Information Retrieval*, pages 137–145, New Orleans, US. ACM Press, New York, US.

Yeh, A., Morgan, A., Colosimo, M., and Hirschman, L. (2005). Biocreative task 1a: gene mention finding evaluation. *BMC Bioinformatics*, 6(Suppl 1):S2.

Young, K. H. (1998). Yeast two-hybrid: so many interactions, (in) so little time.. *Biol Reprod*, 58(2):302–311.

Yuan, Y., Lin, L., Dong, Q., Wang, X., and Li, M. (2005). A protein classification method based on latent semantic analysis. *Engineering in Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the*, pages 7738–7741.

Zheng, B., McLean, D. C., and Lu, X. (2006). Identifying biological concepts from a protein-related corpus with a probabilistic topic model. *BMC Bioinformatics*, 7:58–58.

Appendix A

Tables of Results

The following tables show the full results from the algorithm comparison experiments.

The feature sets are:

- **F1:** Long words with original capitalisation and full word length
- **F2:** Long words with original capitalisation and length truncated to 10
- **F3:** Long words with lowercase, truncated to 10 letters
- **F4:** Long words with lowercase and stemming
- **F5:** Short words with lowercase, truncated to 10 letters
- **F6:** Short words with lower case and stemming

A.1 Gaussian kernel results

		GP			SVM			NB
Data	Features	K	Settings	AUC	K	Settings	AUC	AUC
BC	F1	G	θ=1.0e-02	83.62 $\pm$ 0.45	G	θ=1.0e-01 C=1.0e+00	82.42 $\pm$ 0.50	81.21 $\pm$ 0.58
BC	F3	G	θ=1.0e-02	83.95 $\pm$ 0.43	G	θ=1.0e-01 C=1.0e+00	81.90 $\pm$ 0.49	81.26 $\pm$ 0.52
BC	F2	G	θ=1.0e-02	83.68 $\pm$ 0.48	G	θ=1.0e-01 C=1.0e+00	82.48 $\pm$ 0.54	81.60 $\pm$ 0.60
BC	F4	G	θ=1.0e-02	84.86 $\pm$ 0.46	G	θ=1.0e-01 C=1.0e+01	84.49 $\pm$ 0.47	81.65 $\pm$ 0.56
BC	NER + F1	G	θ=1.0e-02	86.29 $\pm$ 0.41	G	θ=1.0e-01 C=1.0e+00	85.69 $\pm$ 0.45	83.10 $\pm$ 0.52
BC	NER + F3	G	θ=1.0e-02	86.88 $\pm$ 0.40	G	θ=1.0e-02 C=1.0e+00	85.79 $\pm$ 0.48	83.11 $\pm$ 0.51
BC	NER + F2	G	θ=1.0e-02	86.40 $\pm$ 0.44	G	θ=1.0e-01 C=1.0e+00	85.90 $\pm$ 0.46	83.17 $\pm$ 0.58
BC	NER + F4	G	θ=1.0e-02	87.62 $\pm$ 0.42	G	θ=1.0e-01 C=1.0e-01	87.27 $\pm$ 0.43	83.97 $\pm$ 0.48
BC	NER + F5	G	θ=1.0e-02	86.98 $\pm$ 0.44	G	θ=1.0e-02 C=1.0e+00	85.51 $\pm$ 0.50	83.04 $\pm$ 0.56
BC	NER + F6	G	θ=1.0e-02	87.56 $\pm$ 0.38	G	θ=1.0e-01 C=1.0e+01	86.71 $\pm$ 0.45	83.98 $\pm$ 0.54
BC	PROT + F1	G	θ=1.0e-02	91.63 $\pm$ 0.27	G	θ=1.0e-03 C=1.0e-05	91.43 $\pm$ 0.33	85.27 $\pm$ 0.48
BC	PROT + F3	G	θ=1.0e-02	91.81 $\pm$ 0.23	G	θ=1.0e-01 C=1.0e+00	91.55 $\pm$ 0.28	85.96 $\pm$ 0.51
BC	PROT + F2	G	θ=1.0e-02	91.44 $\pm$ 0.26	G	θ=1.0e-03 C=1.0e-04	91.24 $\pm$ 0.30	84.94 $\pm$ 0.52
BC	PROT + F4	G	θ=1.0e-02	92.27 $\pm$ 0.24	G	θ=1.0e-01 C=1.0e+00	92.37 $\pm$ 0.29	86.80 $\pm$ 0.47
BC	PROT + F5	G	θ=1.0e-02	91.78 $\pm$ 0.28	G	θ=1.0e-01 C=1.0e-03	91.42 $\pm$ 0.30	85.84 $\pm$ 0.49
BC	PROT + F6	G	θ=1.0e-02	92.10 $\pm$ 0.27	G	θ=1.0e-01 C=1.0e-03	92.32 $\pm$ 0.26	86.00 $\pm$ 0.43
BC	F5	G	θ=1.0e-02	85.08 $\pm$ 0.42	G	θ=1.0e-01 C=1.0e+00	83.05 $\pm$ 0.45	82.52 $\pm$ 0.50
BC	F6	G	θ=1.0e-02	85.52 $\pm$ 0.40	G	θ=1.0e-01 C=1.0e-01	84.30 $\pm$ 0.44	82.56 $\pm$ 0.48

		GP			SVM			NB
Data	Features	K	Settings	AUC	K	Settings	AUC	AUC
MIPS Abs	F1	G	θ=1.0e-03	86.03 $\pm$ 0.35	G	θ=1.0e-03 C=1.0e+01	93.75 $\pm$ 0.19	55.59 $\pm$ 0.58
MIPS Abs	F3	G	θ=1.0e-03	87.27 $\pm$ 0.28	G	θ=1.0e-03 C=1.0e+01	94.64 $\pm$ 0.19	55.92 $\pm$ 0.62
MIPS Abs	F2	G	θ=1.0e-03	86.15 $\pm$ 0.32	G	θ=1.0e-03 C=1.0e+01	93.41 $\pm$ 0.19	55.94 $\pm$ 0.63
MIPS Abs	F4	G	θ=1.0e-03	89.55 $\pm$ 0.31	G	θ=1.0e-03 C=1.0e+01	95.02 $\pm$ 0.20	55.39 $\pm$ 0.66
MIPS Abs	NER + F1	G	θ=1.0e-03	88.70 $\pm$ 0.35	G	θ=1.0e-03 C=1.0e+01	94.43 $\pm$ 0.25	53.95 $\pm$ 0.57
MIPS Abs	NER + F3	G	θ=1.0e-03	88.86 $\pm$ 0.32	G	θ=1.0e-03 C=1.0e+01	94.12 $\pm$ 0.19	55.73 $\pm$ 0.59
MIPS Abs	NER + F2	G	θ=1.0e-03	88.07 $\pm$ 0.32	G	θ=1.0e-03 C=1.0e+01	93.69 $\pm$ 0.24	56.56 $\pm$ 0.67
MIPS Abs	NER + F4	G	θ=1.0e-03	92.23 $\pm$ 0.25	G	θ=1.0e-03 C=1.0e+01	96.21 $\pm$ 0.17	54.42 $\pm$ 0.65
MIPS Abs	NER + F5	G	θ=1.0e-03	90.06 $\pm$ 0.26	G	θ=1.0e-03 C=1.0e+01	95.22 $\pm$ 0.14	58.56 $\pm$ 0.63
MIPS Abs	NER + F6	G	θ=1.0e-03	92.07 $\pm$ 0.24	G	θ=1.0e-03 C=1.0e+01	95.27 $\pm$ 0.18	55.77 $\pm$ 0.56
MIPS Abs	F5	G	θ=1.0e-03	91.85 $\pm$ 0.23	G	θ=1.0e-03 C=1.0e+01	96.40 $\pm$ 0.11	56.66 $\pm$ 0.58
MIPS Abs	F6	G	θ=1.0e-03	93.03 $\pm$ 0.23	G	θ=1.0e-03 C=1.0e+01	96.21 $\pm$ 0.17	57.03 $\pm$ 0.59

		GP			SVM			NB
Data	Features	K	Settings	AUC	K	Settings	AUC	AUC
MIPS Sent	F1	G	θ=1.0e-02	76.11 $\pm$ 0.41	G	θ=1.0e-02 C=1.0e+01	76.46 $\pm$ 0.43	77.57 $\pm$ 0.40
MIPS Sent	F3	G	θ=1.0e-02	77.81 $\pm$ 0.37	G	θ=1.0e-01 C=1.0e-01	79.12 $\pm$ 0.38	79.12 $\pm$ 0.42
MIPS Sent	F2	G	θ=1.0e-02	75.41 $\pm$ 0.42	G	θ=1.0e-01 C=1.0e+00	76.80 $\pm$ 0.46	75.41 $\pm$ 0.44
MIPS Sent	F4	G	θ=1.0e-02	77.19 $\pm$ 0.33	G	θ=1.0e-02 C=1.0e+01	80.34 $\pm$ 0.32	82.31 $\pm$ 0.34
MIPS Sent	NER + F1	G	θ=1.0e-02	79.11 $\pm$ 0.31	G	θ=1.0e-03 C=1.0e+01	77.25 $\pm$ 0.34	72.68 $\pm$ 0.39
MIPS Sent	NER + F3	G	θ=1.0e-02	78.73 $\pm$ 0.31	G	θ=1.0e-01 C=1.0e+00	75.16 $\pm$ 0.33	70.94 $\pm$ 0.38
MIPS Sent	NER + F2	G	θ=1.0e-02	80.34 $\pm$ 0.30	G	θ=1.0e-01 C=1.0e+00	77.11 $\pm$ 0.34	72.74 $\pm$ 0.41
MIPS Sent	NER + F4	G	θ=1.0e-02	80.28 $\pm$ 0.26	G	θ=1.0e-01 C=1.0e-01	77.14 $\pm$ 0.34	73.51 $\pm$ 0.36
MIPS Sent	NER + F5	G	θ=1.0e-02	83.52 $\pm$ 0.23	G	θ=1.0e-03 C=1.0e+01	82.14 $\pm$ 0.28	77.00 $\pm$ 0.28
MIPS Sent	NER + F6	G	θ=1.0e-02	83.32 $\pm$ 0.28	G	θ=1.0e-03 C=1.0e+00	81.09 $\pm$ 0.30	78.88 $\pm$ 0.34
MIPS Sent	F5	G	θ=1.0e-02	86.90 $\pm$ 0.27	G	θ=1.0e-01 C=1.0e+00	86.53 $\pm$ 0.29	81.86 $\pm$ 0.35
MIPS Sent	F6	G	θ=1.0e-02	85.36 $\pm$ 0.28	G	θ=1.0e-01 C=1.0e+00	85.10 $\pm$ 0.32	83.01 $\pm$ 0.35

		GP			SVM			NB
Data	**Features**	**K**	**Settings**	**AUC**	**K**	**Settings**	**AUC**	**AUC**
PB	F1	G	θ=1.0e-03	92.33 $\pm$ 0.20	G	θ=1.0e-03 C=1.0e+01	91.91 $\pm$ 0.22	54.92 $\pm$ 0.47
PB	F3	G	θ=1.0e-03	92.66 $\pm$ 0.27	G	θ=1.0e-03 C=1.0e+01	92.30 $\pm$ 0.26	53.45 $\pm$ 0.56
PB	F2	G	θ=1.0e-03	92.21 $\pm$ 0.25	G	θ=1.0e-03 C=1.0e+01	91.85 $\pm$ 0.27	54.46 $\pm$ 0.48
PB	F4	G	θ=1.0e-03	92.99 $\pm$ 0.22	G	θ=1.0e-03 C=1.0e+00	92.93 $\pm$ 0.23	53.64 $\pm$ 0.50
PB	NER + F1	G	θ=1.0e-03	90.99 $\pm$ 0.25	G	θ=1.0e-03 C=1.0e+01	91.76 $\pm$ 0.26	67.23 $\pm$ 0.54
PB	NER + F3	G	θ=1.0e-03	91.31 $\pm$ 0.23	G	θ=1.0e-03 C=1.0e+01	91.79 $\pm$ 0.21	66.29 $\pm$ 0.45
PB	NER + F2	G	θ=1.0e-03	91.02 $\pm$ 0.25	G	θ=1.0e-03 C=1.0e+01	91.79 $\pm$ 0.22	66.99 $\pm$ 0.55
PB	NER + F4	G	θ=1.0e-03	92.25 $\pm$ 0.25	G	θ=1.0e-03 C=1.0e+00	92.51 $\pm$ 0.25	66.49 $\pm$ 0.50
PB	NER + F5	G	θ=1.0e-03	91.62 $\pm$ 0.25	G	θ=1.0e-03 C=1.0e+00	91.73 $\pm$ 0.25	62.87 $\pm$ 0.52
PB	NER + F6	G	θ=1.0e-03	92.59 $\pm$ 0.27	G	θ=1.0e-03 C=1.0e+00	92.83 $\pm$ 0.26	61.58 $\pm$ 0.55
PB	F5	G	θ=1.0e-03	92.72 $\pm$ 0.25	G	θ=1.0e-03 C=1.0e+00	92.21 $\pm$ 0.26	52.41 $\pm$ 0.51
PB	F6	G	θ=1.0e-03	93.34 $\pm$ 0.25	G	θ=1.0e-03 C=1.0e+00	93.26 $\pm$ 0.25	52.81 $\pm$ 0.52

		GP			SVM			NB
Data	Features	K	Settings	AUC	K	Settings	AUC	AUC
AImed	F1	G	θ=1.0e-02	82.83 $\pm$ 0.30	G	θ=1.0e-02 C=1.0e+01	84.10 $\pm$ 0.31	81.50 $\pm$ 0.35
AImed	F3	G	θ=1.0e-02	83.51 $\pm$ 0.31	G	θ=1.0e-02 C=1.0e+01	84.56 $\pm$ 0.29	81.86 $\pm$ 0.33
AImed	F2	G	θ=1.0e-02	82.89 $\pm$ 0.31	G	θ=1.0e-02 C=1.0e+01	84.16 $\pm$ 0.28	81.79 $\pm$ 0.33
AImed	F4	G	θ=1.0e-02	83.25 $\pm$ 0.33	G	θ=1.0e-02 C=1.0e+01	84.11 $\pm$ 0.34	81.36 $\pm$ 0.34
AImed	NER + F1	G	θ=1.0e-02	88.13 $\pm$ 0.22	G	θ=1.0e-02 C=1.0e+01	89.52 $\pm$ 0.23	86.65 $\pm$ 0.28
AImed	NER + F3	G	θ=1.0e-02	88.49 $\pm$ 0.23	G	θ=1.0e-02 C=1.0e+01	89.89 $\pm$ 0.22	87.10 $\pm$ 0.25
AImed	NER + F2	G	θ=1.0e-02	88.08 $\pm$ 0.24	G	θ=1.0e-02 C=1.0e+01	89.34 $\pm$ 0.24	86.42 $\pm$ 0.26
AImed	NER + F4	G	θ=1.0e-02	89.25 $\pm$ 0.22	G	θ=1.0e-02 C=1.0e+01	89.62 $\pm$ 0.22	87.11 $\pm$ 0.22
AImed	NER + F5	G	θ=1.0e-02	88.34 $\pm$ 0.25	G	θ=1.0e-02 C=1.0e+01	89.70 $\pm$ 0.23	86.70 $\pm$ 0.28
AImed	NER + F6	G	θ=1.0e-02	83.28 $\pm$ 0.30	G	θ=1.0e-01 C=1.0e+00	83.41 $\pm$ 0.28	79.45 $\pm$ 0.33
AImed	PROT + F1	G	θ=1.0e-01	89.57 $\pm$ 0.23	G	θ=1.0e-01 C=1.0e+00	90.46 $\pm$ 0.22	83.33 $\pm$ 0.30
AImed	PROT + F3	G	θ=1.0e-01	89.85 $\pm$ 0.29	G	θ=1.0e-01 C=1.0e+00	90.91 $\pm$ 0.25	83.65 $\pm$ 0.31
AImed	PROT + F2	G	θ=1.0e-01	89.85 $\pm$ 0.20	G	θ=1.0e-01 C=1.0e+00	90.71 $\pm$ 0.19	83.44 $\pm$ 0.28
AImed	PROT + F4	G	θ=1.0e-02	90.24 $\pm$ 0.20	G	θ=1.0e-01 C=1.0e+00	91.18 $\pm$ 0.22	83.60 $\pm$ 0.29
AImed	PROT + F5	G	θ=1.0e-01	89.85 $\pm$ 0.22	G	θ=1.0e-01 C=1.0e+00	90.97 $\pm$ 0.21	83.80 $\pm$ 0.31
AImed	PROT + F6	G	θ=1.0e-02	90.20 $\pm$ 0.18	G	θ=1.0e-01 C=1.0e+00	91.20 $\pm$ 0.20	83.74 $\pm$ 0.32
AImed	F5	G	θ=1.0e-02	83.85 $\pm$ 0.28	G	θ=1.0e-02 C=1.0e+01	84.65 $\pm$ 0.27	81.12 $\pm$ 0.32
AImed	F6	G	θ=1.0e-02	83.56 $\pm$ 0.26	G	θ=1.0e-02 C=1.0e+01	84.10 $\pm$ 0.25	80.37 $\pm$ 0.26

A.2 Cosine kernel results

		GP		SVM			VBpMKL	
Data	Features	K	AUC	K	Settings	AUC	K	AUC
BC	F1	C	74.93 ± 0.61	C	C=1.0e-01	80.94 ± 0.51	C	75.33 ± 0.64
BC	F3	C	75.20 ± 0.60	C	C=1.0e-01	80.61 ± 0.55	C	76.06 ± 0.61
BC	F2	C	74.72 ± 0.64	C	C=1.0e-02	81.01 ± 0.63	C	75.44 ± 0.64
BC	F4	C	77.75 ± 0.65	C	C=1.0e-01	82.39 ± 0.56	C	79.03 ± 0.65
BC	NER + F1	C	76.63 ± 0.64	C	C=1.0e+00	85.85 ± 0.47	C	76.87 ± 0.63
BC	NER + F3	C	76.74 ± 0.59	C	C=1.0e+00	85.41 ± 0.46	C	77.23 ± 0.59
BC	NER + F2	C	76.65 ± 0.66	C	C=1.0e+00	85.97 ± 0.49	C	76.63 ± 0.68
BC	NER + F4	C	78.70 ± 0.59	C	C=1.0e-01	87.23 ± 0.45	C	80.04 ± 0.58
BC	NER + F5	C	76.39 ± 0.66	C	C=1.0e+00	84.94 ± 0.49	C	76.92 ± 0.65
BC	NER + F6	C	78.47 ± 0.61	C	C=1.0e-01	86.26 ± 0.44	C	79.83 ± 0.60
BC	PROT + F1	C	83.67 ± 0.50	C	C=1.0e+00	91.37 ± 0.32	C	83.57 ± 0.50
BC	PROT + F3	C	84.61 ± 0.42	C	C=1.0e+00	91.50 ± 0.27	C	84.57 ± 0.43
BC	PROT + F2	C	83.46 ± 0.47	C	C=1.0e+00	91.00 ± 0.28	C	83.28 ± 0.47
BC	PROT + F4	C	86.57 ± 0.47	C	C=1.0e+00	92.45 ± 0.28	C	86.96 ± 0.46
BC	PROT + F5	C	84.10 ± 0.50	C	C=1.0e-04	91.42 ± 0.31	C	84.28 ± 0.49
BC	PROT + F6	C	86.02 ± 0.43	C	C=1.0e+00	92.17 ± 0.28	C	86.43 ± 0.42
BC	F5	C	75.75 ± 0.59	C	C=1.0e-01	81.48 ± 0.53	C	76.81 ± 0.56
BC	F6	C	78.19 ± 0.56	C	C=1.0e-01	82.89 ± 0.51	C	79.69 ± 0.53

		GP		SVM			VBpMKL	
Data	Features	K	AUC	K	Settings	AUC	K	AUC
MIPS Abs	F1	C	89.54 ± 0.29	C	C=1.0e+01	92.25 ± 0.25	C	89.60 ± 0.28
MIPS Abs	F3	C	91.99 ± 0.20	C	C=1.0e+01	93.86 ± 0.22	C	92.63 ± 0.17
MIPS Abs	F2	C	90.73 ± 0.22	C	C=1.0e+01	92.08 ± 0.24	C	90.69 ± 0.21
MIPS Abs	F4	C	92.03 ± 0.23	C	C=1.0e+01	92.77 ± 0.32	C	91.80 ± 0.23
MIPS Abs	NER + F1	C	91.63 ± 0.27	C	C=1.0e+00	95.20 ± 0.20	C	92.45 ± 0.26
MIPS Abs	NER + F3	C	91.21 ± 0.28	C	C=1.0e+00	94.77 ± 0.19	C	91.81 ± 0.25
MIPS Abs	NER + F2	C	90.74 ± 0.25	C	C=1.0e+00	94.77 ± 0.19	C	91.68 ± 0.23
MIPS Abs	NER + F4	C	93.45 ± 0.25	C	C=1.0e+01	96.60 ± 0.13	C	93.67 ± 0.24
MIPS Abs	NER + F5	C	92.19 ± 0.24	C	C=1.0e+02	95.84 ± 0.14	C	92.79 ± 0.22
MIPS Abs	NER + F6	C	93.39 ± 0.19	C	C=1.0e+01	94.96 ± 0.20	C	93.59 ± 0.19
MIPS Abs	F5	C	94.87 ± 0.16	C	C=1.0e+00	95.76 ± 0.16	C	95.53 ± 0.14
MIPS Abs	F6	C	95.31 ± 0.15	C	C=1.0e+01	95.22 ± 0.21	C	95.38 ± 0.16

		GP		SVM			VBpMKL	
Data	Features	K	AUC	K	Settings	AUC	K	AUC
MIPS Sent	F1	C	69.55 $\pm$ 0.43	C	C=1.0e+00	75.95 $\pm$ 0.40	C	68.84 $\pm$ 0.43
MIPS Sent	F3	C	72.50 $\pm$ 0.43	C	C=1.0e+00	77.94 $\pm$ 0.37	C	71.98 $\pm$ 0.42
MIPS Sent	F2	C	69.21 $\pm$ 0.46	C	C=1.0e+00	75.18 $\pm$ 0.48	C	69.14 $\pm$ 0.45
MIPS Sent	F4	C	73.29 $\pm$ 0.38	C	C=1.0e+00	79.35 $\pm$ 0.32	C	71.63 $\pm$ 0.41
MIPS Sent	NER + F1	C	69.70 $\pm$ 0.39	C	C=1.0e-05	77.70 $\pm$ 0.34	C	69.32 $\pm$ 0.39
MIPS Sent	NER + F3	C	66.00 $\pm$ 0.39	C	C=1.0e-05	75.06 $\pm$ 0.34	C	65.32 $\pm$ 0.39
MIPS Sent	NER + F2	C	70.50 $\pm$ 0.40	C	C=1.0e-05	77.69 $\pm$ 0.36	C	70.09 $\pm$ 0.41
MIPS Sent	NER + F4	C	72.23 $\pm$ 0.39	C	C=1.0e-05	78.82 $\pm$ 0.29	C	71.47 $\pm$ 0.38
MIPS Sent	NER + F5	C	75.06 $\pm$ 0.34	C	C=1.0e-05	81.14 $\pm$ 0.28	C	75.33 $\pm$ 0.34
MIPS Sent	NER + F6	C	75.60 $\pm$ 0.38	C	C=1.0e-05	80.31 $\pm$ 0.30	C	75.47 $\pm$ 0.39
MIPS Sent	F5	C	80.43 $\pm$ 0.36	C	C=1.0e+00	83.23 $\pm$ 0.33	C	80.81 $\pm$ 0.37
MIPS Sent	F6	C	80.76 $\pm$ 0.36	C	C=1.0e+00	82.73 $\pm$ 0.33	C	80.47 $\pm$ 0.35

		GP		SVM			VBpMKL	
Data	Features	K	AUC	K	Settings	AUC	K	AUC
PB	F1	C	92.38 $\pm$ 0.20	C	C=1.0e+00	92.17 $\pm$ 0.21	C	92.34 $\pm$ 0.21
PB	F3	C	92.73 $\pm$ 0.26	C	C=1.0e+00	92.58 $\pm$ 0.26	C	92.52 $\pm$ 0.26
PB	F2	C	92.32 $\pm$ 0.25	C	C=1.0e+00	92.04 $\pm$ 0.26	C	92.33 $\pm$ 0.24
PB	F4	C	92.83 $\pm$ 0.22	C	C=1.0e+00	92.84 $\pm$ 0.22	C	92.74 $\pm$ 0.22
PB	NER + F1	C	91.97 $\pm$ 0.24	C	C=1.0e+00	92.21 $\pm$ 0.23	C	92.12 $\pm$ 0.25
PB	NER + F3	C	92.12 $\pm$ 0.22	C	C=1.0e+00	92.39 $\pm$ 0.22	C	92.18 $\pm$ 0.22
PB	NER + F2	C	92.01 $\pm$ 0.23	C	C=1.0e+00	92.30 $\pm$ 0.23	C	92.16 $\pm$ 0.23
PB	NER + F4	C	92.37 $\pm$ 0.26	C	C=1.0e+00	92.77 $\pm$ 0.26	C	92.58 $\pm$ 0.26
PB	NER + F5	C	92.11 $\pm$ 0.23	C	C=1.0e+00	92.32 $\pm$ 0.24	C	92.16 $\pm$ 0.23
PB	NER + F6	C	92.58 $\pm$ 0.26	C	C=1.0e+00	93.07 $\pm$ 0.25	C	92.86 $\pm$ 0.25
PB	F5	C	92.63 $\pm$ 0.24	C	C=1.0e+00	92.33 $\pm$ 0.24	C	92.44 $\pm$ 0.23
PB	F6	C	93.02 $\pm$ 0.25	C	C=1.0e+00	93.02 $\pm$ 0.25	C	93.15 $\pm$ 0.24

		GP		SVM			VBpMKL	
Data	Features	K	AUC	K	Settings	AUC	K	AUC
AImed	F1	C	80.84 $\pm$ 0.33	C	C=1.0e+00	83.22 $\pm$ 0.31	C	80.56 $\pm$ 0.32
AImed	F3	C	81.92 $\pm$ 0.33	C	C=1.0e+00	84.17 $\pm$ 0.30	C	81.39 $\pm$ 0.34
AImed	F2	C	80.93 $\pm$ 0.34	C	C=1.0e+00	83.30 $\pm$ 0.30	C	80.71 $\pm$ 0.35
AImed	F4	C	81.67 $\pm$ 0.36	C	C=1.0e+00	83.39 $\pm$ 0.32	C	81.02 $\pm$ 0.35
AImed	NER + F1	C	87.42 $\pm$ 0.27	C	C=1.0e+00	90.09 $\pm$ 0.22	C	87.13 $\pm$ 0.26
AImed	NER + F3	C	88.03 $\pm$ 0.25	C	C=1.0e+00	90.51 $\pm$ 0.21	C	87.71 $\pm$ 0.26
AImed	NER + F2	C	87.28 $\pm$ 0.27	C	C=1.0e+00	89.87 $\pm$ 0.23	C	86.91 $\pm$ 0.28
AImed	NER + F4	C	88.36 $\pm$ 0.23	C	C=1.0e+00	90.28 $\pm$ 0.21	C	88.04 $\pm$ 0.24
AImed	NER + F5	C	87.76 $\pm$ 0.28	C	C=1.0e+00	90.36 $\pm$ 0.22	C	87.55 $\pm$ 0.27
AImed	NER + F6	C	80.34 $\pm$ 0.27	C	C=1.0e+00	83.88 $\pm$ 0.26	C	87.86 $\pm$ 0.24
AImed	PROT + F1	C	87.15 $\pm$ 0.25	C	C=1.0e+00	90.92 $\pm$ 0.21	C	86.49 $\pm$ 0.27
AImed	PROT + F3	C	87.69 $\pm$ 0.28	C	C=1.0e+00	91.38 $\pm$ 0.24	C	87.17 $\pm$ 0.28
AImed	PROT + F2	C	87.40 $\pm$ 0.23	C	C=1.0e+00	91.05 $\pm$ 0.18	C	86.74 $\pm$ 0.24
AImed	PROT + F4	C	87.83 $\pm$ 0.26	C	C=1.0e+00	91.32 $\pm$ 0.20	C	87.36 $\pm$ 0.26
AImed	PROT + F5	C	87.80 $\pm$ 0.25	C	C=1.0e+00	91.49 $\pm$ 0.19	C	87.17 $\pm$ 0.27
AImed	PROT + F6	C	87.84 $\pm$ 0.25	C	C=1.0e+00	91.47 $\pm$ 0.19	C	87.32 $\pm$ 0.26
AImed	F5	C	81.77 $\pm$ 0.30	C	C=1.0e+00	84.09 $\pm$ 0.27	C	81.34 $\pm$ 0.30
AImed	F6	C	81.63 $\pm$ 0.26	C	C=1.0e+00	83.64 $\pm$ 0.24	C	81.07 $\pm$ 0.26

A.3 Kernel combination results

BC

Settings	VBpMKL
combination:	F=0.5209 ± 0.0117
convex linear	E=13.6928 ± 0.3641
$\sum_{L=1}^{15} \mathbf{X}\kappa(\mathbf{H}_L, \mathbf{H}_L)\mathbf{X}^T$	P=0.6575 ± 0.0137
kernel: Gaussian	R=0.4464 ± 0.0130
	A=0.9251 ± 0.0027
combination:	F=0.6326 ± 0.0099
uniform	E=12.1339 ± 0.3148
$\sum_{L=1}^{15} \mathbf{X}\kappa(\mathbf{H}_L, \mathbf{H}_L)\mathbf{X}^T$	P=0.6625 ± 0.0125
kernel: Gaussian	R=0.6230 ± 0.0127
	A=0.9134 ± 0.0038
combination:	F=0.4249 ± 0.0120
convex linear	E=14.9564 ± 0.3175
$\sum_{l=1}^{15} \mathbf{X}\kappa(\mathbf{H}_l, \mathbf{H}_l)\mathbf{X}^T$	P=0.6352 ± 0.0163
kernel: Gaussian	R=0.3345 ± 0.0119
	A=0.9126 ± 0.0026
combination:	F=0.6165 ± 0.0088
uniform	E=12.1632 ± 0.3243
$\sum_{l=1}^{15} \mathbf{X}\kappa(\mathbf{H}_l, \mathbf{H}_l)\mathbf{X}^T$	P=0.6799 ± 0.0109
kernel: Gaussian	R=0.5762 ± 0.0113
	A=0.9288 ± 0.0027
combination:	F=0.5094 ± 0.0102
convex linear	E=14.0352 ± 0.3622
$\mathbf{B}_s(OAA)$ $\mathbf{B}_g(OAA)$	P=0.6552 ± 0.0146
$\mathbf{B}_s(GEN)$ $\mathbf{B}_g(GEN)$	R=0.4316 ± 0.0109
kernel: Gaussian	A=0.9168 ± 0.0028
combination:	F=0.6217 ± 0.0105
uniform	E=11.6227 ± 0.3225
$\mathbf{B}_s(OAA)$ $\mathbf{B}_g(OAA)$	P=0.7048 ± 0.0114
$\mathbf{B}_s(GEN)$ $\mathbf{B}_g(GEN)$	R=0.5720 ± 0.0131
kernel: Gaussian	A=0.9339 ± 0.0024
combination:	F=0.6019 ± 0.0092
uniform	E=12.3635 ± 0.3259
$\mathbf{H}_{L=12}$ (weighted sum)	P=0.6793 ± 0.0132
$\mathbf{B}_s(OAA)$	R=0.5559 ± 0.0110
kernel: Gaussian	A=0.9270 ± 0.0026
combination: uniform	F=0.6351 ± 0.0099
$\mathbf{B}_s(OAA)$ $\mathbf{B}_g(OAA)$	E=11.5616 ± 0.3395
$\mathbf{B}_s(GEN)$ $\mathbf{B}_g(GEN)$	P=0.7046 ± 0.0112
$\mathbf{H}_{L=12}$ (weighted sum)	R=0.5937 ± 0.0131
kernel: Gaussian	A=0.9333 ± 0.0027

Almed

Settings	VBpMKL
combination:	F=0.6884 ± 0.0069
convex linear	E=17.5912 ± 0.2975
$\sum_{L=1}^{15} \mathbf{X}\kappa(\mathbf{H}_L, \mathbf{H}_L)\mathbf{X}^T$	P=0.7631 ± 0.0067
kernel: Gaussian	R=0.6401 ± 0.0106
	A=0.9019 ± 0.0026
combination:	F=0.7080 ± 0.0043
uniform	E=18.4865 ± 0.2735
$\sum_{L=1}^{15} \mathbf{X}\kappa(\mathbf{H}_L, \mathbf{H}_L)\mathbf{X}^T$	P=0.6950 ± 0.0058
kernel: Gaussian	R=0.7253 ± 0.0053
	A=0.8816 ± 0.0024
combination:	F=0.6808 ± 0.0088
convex linear	E=18.2586 ± 0.3242
$\sum_{l=1}^{15} \mathbf{X}\kappa(\mathbf{H}_l, \mathbf{H}_l)\mathbf{X}^T$	P=0.7393 ± 0.0063
kernel: Gaussian	R=0.6438 ± 0.0103
	A=0.8975 ± 0.0025
combination:	F=0.7216 ± 0.0044
uniform	E=17.2270 ± 0.2360
$\sum_{l=1}^{15} \mathbf{X}\kappa(\mathbf{H}_l, \mathbf{H}_l)\mathbf{X}^T$	P=0.7223 ± 0.0052
kernel: Gaussian	R=0.7243 ± 0.0060
	A=0.8982 ± 0.0021
combination:	F=0.7048 ± 0.0073
convex linear	E=17.3179 ± 0.3007
$\mathbf{B}_s(OAA)$ $\mathbf{B}_g(OAA)$	P=0.7442 ± 0.0064
$\mathbf{B}_s(GEN)$ $\mathbf{B}_g(GEN)$	R=0.6820 ± 0.0101
kernel: Gaussian	A=0.9051 ± 0.0023
combination:	F=0.7414 ± 0.0039
uniform	E=15.9214 ± 0.2291
$\mathbf{B}_s(OAA)$ $\mathbf{B}_g(OAA)$	P=0.7474 ± 0.0056
$\mathbf{B}_s(GEN)$ $\mathbf{B}_g(GEN)$	R=0.7395 ± 0.0054
kernel: Gaussian	A=0.9105 ± 0.0019
combination:	F=0.7287 ± 0.0042
uniform	E=16.5779 ± 0.2388
$\mathbf{H}_{L=11}$ (uniform sum)	P=0.7387 ± 0.0055
$\mathbf{B}_s(OAA)$	R=0.7224 ± 0.0054
kernel: Gaussian	A=0.9037 ± 0.0021
combination: uniform	F=0.7424 ± 0.0041
$\mathbf{B}_s(OAA)$ $\mathbf{B}_g(OAA)$	E=15.8507 ± 0.2528
$\mathbf{B}_s(GEN)$ $\mathbf{B}_g(GEN)$	P=0.7491 ± 0.0057
$\mathbf{H}_{L=11}$ (uniform sum)	R=0.7399 ± 0.0058
kernel: Gaussian	A=0.9113 ± 0.0021